PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

MAESTRÍA EN REDES DE COMUNICACIÓN

INFORME FINAL CASO DE ESTUDIO PARA UNIDAD DE TITUALCIÓN ESPECIAL

TEMA:

DISEÑO Y CONSTRUCCIÓN DE UN PROTOTIPO QUE PERMITA LA GEOLOCALIZACIÓN DE PUNTOS MEDIANTE LA UTILIZACIÓN DE REDES GPS A TRAVÉS DE CONEXIONES MÓVILES DE DATOS 3G-4G-GPRS.

Olmedo Xavier Narváez Montenegro

Quito – 2016

AUTORÍA

Yo, Olmedo Xavier Narváez Montenegro, portador de la cédula de ciudadanía No. 0401375035, declaro bajo juramento que la presente investigación es de total responsabilidad del autor, y que se he respetado las diferentes fuentes de información con las citas correspondientes. Esta investigación no contiene plagio alguno y es resultado de un trabajo serio desarrollado en su totalidad por mi persona.

_____

Olmedo Xavier Narváez Montenegro

INDICE DE CONTENIDO

INDICE DE ILUSTRACIONES

INDICE DE TABLAS

CAPITULO I

GENERALIDADES

1.1 INTRODUCCIÓN

En la actualidad se vive en un era donde la tecnológia ha tomado un papel muy importante para el desarrollo de las actividades diarias.La penetración en el mundo y en la vida diaria de los denominados smartphones o dispositivos inteligentes, y la paralela generalización de tarifas de conexión a Internet a precios realmente asequibles, se convienten en una oportunidad para el desarrollo de las denominadas aplicaciones móviles en sistemas operativos OpenSource, como Android, así como la propagación de estas ya que ofrecen un abanico de servicios para facilitar la interconexión en distintos puntos del mundo entre los usuarios.

En el presente trabajo se plantea elaborar el Diseño y Construcción de un prototipo que permita la geolocalización de puntos mediante la utilización de redes GPS a través de conexiones móviles de datos 3g-4g-gprs.

El proyecto tuvo como finalidad desarrollar un prototipo para la automatización de información mediante la utilización de una aplicación móvil desarrollada en Android, utilizando un IDE de desarrollo llamado Android Studio. La aplicación desarrollada permite el levantamiento, captura y manejo de datos en campo. Esta acción se realiza mediante la determinación de la posición de un elemento en un sistema de coordenadas utilizando el posicionamiento a través de la red GPS, que mediante el uso de herramientas tecnológicas se transfiere, transforma, procesa y se muestra la información levantada, geográficamente referenciada y automáticamente posicionada dentro de un mapa que se encuentra en el sistema de información turística, mismo que a través de un navegador web se podrá visualizar y consultar por medio de las librerías de Google Maps .

Teniendo en cuenta el crecimiento que está teniendo el mercado de aplicaciones móviles,el proyecto se enfoca en desarrollar una aplicación móvil específicamente para entorno

Android,sin duda, esto obliga, como profesionales del sector tecnológico, a conocer los retos y posibilidades de este entorno, para lo cual se hace necesario realizar un estudio del mismo así como de las herramientas para su desarrollo para luego mediante la selección de la metodología más adecuada, empezar el desarrollo de la aplicación cumpliendo con cada una de sus fases.

Este proyecto deberá proporcionar un manejo eficaz y eficiente en la recogida de datos en campo en comparación a uno operado manualmente, marcando una notable evolución tecnológica en el manejo de datos.

El alcance de este proyecto se presentó tanto en tiempo de duración, como en la inversión, por lo que se decidió desarrollar una aplicación móvil eficiente de bajo costo y funcional a corto plazo, que permita optimizar recursos económicos, humanos, tecnológicos y sobre todo se disponga de información actualizada en menor tiempo.

Esta investigación se divide en cinco capítulos, en el primero se realiza una descrición de los aspectos generales que permiten justificar la necesidad del desarrollo de una aplicación móvil para georeferenciación, así como los acontecimientos relevantes que dan una visión retrospectiva sobre el tema y los objetivos tanto generales como específicios que persigue el caso de estudio en mención.

El capítulo dos se dedica al levantamiento de un diagnóstico sobre las diferentes alternativas existentes para el desarrollo de la aplicación móvil, se profundiza el análisis en el sistema operativo Android asi como también se analiza la aplicación google para geolocalización y mensajería instantánea, GPS y los protocolos de comunicación.

En el capítulo tres se realizó el análisis, diseño e implementación de la app móvil utilizando para su desarrollo el API Android Studio después de haber realizado un análisis de las distintas opciones, permitiéndo observar que la aplicación móvil permitirá entre otros usos:

- Levantar información en campo de forma ágil y rápida.
- Los datos levantados in situ se guardarán automaticamnete una base de datos.

- Facilitará al usuario la obtención de información turística inmediata con solo ingresar al link: http//:www.carchi.gob.ec/turistico donde se mostrará la información levantada en campo de forma ordenada y didactica en cuanto a hoteles, restaurantes, lugares que visitar, de dónde transportarse, sitios turísticos brindando información al viajero durante su estadía en algún punto de la Provincia.
- Observar la posisicón en la que se encuentran las personas que levantan la información in situ, mediante un mapa (Google Maps) donde se puede conocer la ubicación en tiempo real.
- Proporciona una opción de chat la cual realiza comunicación en tiempo real entre el equipo que realiza el levantamiento de información desde distintos puntos de la provincia.

Dentro del capítulo cuatro se procede a plantear los requisitos para la instalación de la app, se dio a conocer su funcionamiento y para comprobar su usabilidad y funcionalidad se ejecutaron diferentes pruebas mismas que permitieron evidenciar la facilidad o complejidad para manipular la app, se verificó que la app desarrollada funciona en varios equipos con versiones diferentes del sistema operativo Android, asi como también se comparó el desempeño de la app en una red Wi-Fi y en un red móvil y los tiempos que estas utilizan, asi como el consumo de megas utilizadas por la app en el uso de sus diferentes componetes.

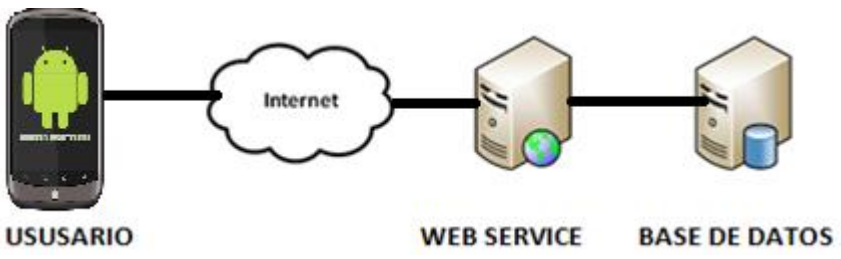
En el capítulo cinco se establecieron conlusiones y recomendaciones mismas que permitieron comprobar el cumplimiento de los objetivos planteados en este caso de estudio.

1.2 JUSTIFICACIÓN

Como una medida para apoyar el desarrollo económico de la Provincia se ha planteado una estrategia de levantamiento y difusión de los atractivos turísticos del Carchi a través de una herramienta de utilidad que facilite el trabajo de levantamiento de información a los actores provinciales de turismo, el fácil acceso para los ciudadanos y automatice los datos turísticos de la Provincia levantados en campo a través de dispositivos móviles Smartphones, lo que permite integrarse de una manera ágil a las plataformas que tiene desarrollado el Gobierno Provincial, propiciando el dialogo armónico con miras a la diversificación y consolidación de información que permita obtener productos turísticos competitivos, así como también se constituya en un elemento de trascendencia para la toma de decisiones tanto para el sector público como para el privado en los niveles nacional, regional y local.

Con el proyecto se buscó principalmente automatizar procesos en la gestión de levantamiento de información turística in situ que se vienen realizando de forma manual, para lo cual se desarrolló una aplicación para dispositivos móviles con sistema operativo Android desdela versión 4.1 en adelante, la misma que interactua con un Servicio Web (WebService) que esta disponible para la comunicación entre el servidor de la aplicación web de turismo y la app móvil, mismo que permite que la aplicación almacene la información levantada automáticamente en la base de datos, desde cualquier lugar de la provincia del Carchi gracias al internet.

Ilustración 1. Proceso de la aplicación móvil



Elaborado por: El Autor

Esta es una de las muchas maneras que se puede aplicar el servicio de la aplicación ya que se podrá reprogramar para que sea utilizada en otros tipos de levantamiento de información simplemente cambiando los formularios y direccionamiento de los mismos.

La importancia de este proyecto radica en la utilización de la tecnología en el desarrollo de una aplicación móvil que sirve para:

- automatizar el proceso
- levantar, capturar y manejar los datos de manera ágil y eficiente
- transferir, transformar, procesar y mostrar la información levantada, geográficamente referenciada y automáticamente posicionada dentro de un mapa a través de un navegador web

Todo esto permite realizar un manejo eficaz y eficiente en la recogida de datos en campo en comparación a uno operado manualmente, marcando una notable evolución tecnológica en el manejo de datos, lo cuál permite obtener un ahorro de tiempo, recursos y dinero; además de ejercer un control del personal asignado a este levantamiento ya que se puede supervisar sus posiciones en tiempo real además de dar disposiciones en un chat que se encuentra integrado en la aplicación.

1.3 ANTECEDENTES

En los últimos años la penetración de la tecnología en diferentes aspectos de la vida cotidiana ha permitido que equipos y sistemas que antes eran restrictivos, hoy se conviertan en herramientas de uso diario para las actividades laborales y de ocio. La tecnología aporta grandes beneficios a la humanidad, su papel principal es crear mejores herramientas útiles para simplificar el ahorro de tiempo y esfuerzo de trabajo, la tecnología juega un papel primordial en el entorno social ya que gracias a ella se podrá realizar la comunicación a cualquier parte de mundo de forma inmediata.Se avanza hacia una comunicación casi sin límites, aparecen las centrales digitales y la fibra óptica, que ofrecen cada vez más servicios a los clientes, y comienza la telefonía móvil.

"En Ecuador las comunicaciones móviles sin duda alguna han experimentado un enorme crecimiento desarrollándose diversas tecnologías y sistemas para dar servicios de comunicación inalámbrica. En el Ecuador el servicio móvil celular inicia a finales de 1993 con la entrada en el mercado de CONECEL S.A. (Porta Celular, luego CLARO) y OTECEL S.A. (al inicio Bellsouth y actualmente denominada Movistar), manteniéndose el dominio de estas 2 empresas hasta el año 2003 cuando entró en operación una tercera operadora TELECSA (al inicio Alegro actualmente CNT E.P.)" Tomado de [1]

Según las Estadísticas de Telecomunicaciones presentadas por la Agencia de Regulación y Control de Telecunicaciones (ARCOTEL) en el año 2015 el porcentaje de participación de cada una de las empresas telefónicas antes mencionadas de acuerdo a la densidad de abonados y a las líneas activas es: CONECEL S.A. 62,48% seguido de OTECEL S.A CON EL 29,83% y CNY S.A. 7,68%.

En el pasado las empresas de telecomunicaciones brindaban un solo servicio: telefonía, audio y video por suscripción, servicios portadores y servicios de valor agregado. En la actualidad un mismo proveedor de servicios dentro de una misma infraestructura de telecomunicaciones, puede brindar múltiples servicios. El avance de las telecomunicaciones en los últimos años y su difusión han permitido dejar de soñar con un mundo

interconectado que puede interoperar y transformarse en una realidad. La telefonía móvil sin duda ha revolucionado el mundo contando desde los primeros teléfonos portátiles instalados en los automóviles hasta los actuales smartphones, permitiendo que la comunicación sea instantánea, permanente y global.

Actualmente, dada la evolución tecnológica del dispositivo electrónico junto con los avances en los estándares tecnológicos de las redes digitales de intercambio, se instala el teléfono móvil como símbolo de la convergencia digital que proporciona un amplio abanico de aplicaciones, funciones y servicios dentro de un único terminal y empleando una diversidad de formatos iguales a las de una computadora, dicha convergencia digital implica pasar desde su tradicional función de llamadas de voz y de enviar mensajes cortos de texto, SMS, a la integración con otros medios que hacen posible el concepto de teléfonos inteligentes o Smartphone los cuales presentan una amplia oferta de servicios como el email, publicidad, información de mercados, transferencias bancarias, realizar compras, enviar y recibir correos electrónicos, escuchar música, ver videos, acceder a redes sociales gestión de datos, ocio electrónico, información meteorológica y de tránsito.

La aceptación de estos aparatos móviles en el mercado ha permitido que las empresas líderes en tecnología vean atractiva esta plaza, direccionen su trabajo y pongan a disposición para teléfonos inteligentes y numerosas aplicaciones dandole un valor agregado para quienes buscan soluciones personales, emprearíales y de entretenimiento en dispositivos móviles satisfaciendo así al usuario.

Según las Estadísticas de Telecomunicaciones presentadas por la Agencia de Regulación y Control de Telecunicaciones (ARCOTEL) para el año 2015 el Ecuador cuenta con 13.859,020 líneas Activas de teléfonos móviles de los cuales 9.252,920 líneas usan tecnología GSM, 3.019,889 líneas usan tecnología UMTS, 636.488 líneas usan tecnología HSPA+ y 949.723 líneas usan tecnología LTE.

Adicionalmente el ARCOTEL señala que del total de líneas activas en el Ecuador se encuentran activadas y brindando solo el servicio para telefonía 7.710,845 líneas, para

telefonía e internet 5.232,946 líneas, 460.322 líneas que solo contraten el servicio de internet y aquellas que solo contratan paquetes de datos ascienden 454.907 líneas.

La misma fuente señala que hasta el año 2015 existen alrededor de 271 radiobases en el país de los cuales 126 pertenecen a la tecnología GSM 850, 36 a tecnología GSM 1900, 63 con tecnología UMTS 850, 40 con tecnología UMTS 1900 y 6 a tecnología LTE (AWS).

Con los datos antes mencionados se evidencia que el uso de la tecnología móvil se ha generalizado con el pasar del tiempo convierteindose en un medio que permite los usuarios interconectados de Ecuador y el mundo acceder a información e interconexión en diferentes partes del mundo con un solo clik a través de la web utilizando el principal motor de búsqueda de internet: Google.

La ejecución de todas las aplicaciones se realiza dentro de un ecosistema donde se presentan varios factores que lo afectan como son: la infraestructura de la aplicación, el sistema operativo, los métodos de entrada de información, los propios usuarios, los canales de distribución de la aplicación, entre otros.

Para el desarrollo de las aplicaciones móviles, el ecosistema es aún más heterogéneo que en el resto de desarrollos ya que pueden ser ejecutadas en diferentes tipos de dispositivo móvil, antiguo o en un moderno ya sea un Smartphone o unaTablet mismos que están diseñados bajo una plataforma informática y en diferentes sistemas operativos (según la marca del dispositivo), por tanto es indispensable conocerlos mas profucndamente para obtener y desarrollo óptimo.

1.4 OBJETIVOS

1.4.1 Objetivo General:

Diseñar y construir una aplicación para dispositivos móviles que permita la geolocalización de puntos mediante la utilización de redes GPS a través de conexiones Wi-Fi y móviles de datos 3g-4g-gprs.

1.4.2 Objetivos Específicos:

a) Levantar un diagnóstico sobre las diferentes alternativas existentes para el desarrollo de aplicaciones móviles.

b) Realizar un análisis de las herramientas de google para la geolocalización y mensajería instantánea de puntos así como del funcionamiento de las conexiones GPS integradas con los dispositivos Android.

c) Realizar el análisis, diseño e implementación de un prototipo funcional para dispositivos Android basado en el IDE Android Studio.

d) Ejecutar pruebas para comprobar la usabilidad y funcionalidad de la aplicación desarrolla.

CAPITULO II

DESARROLLO CASO DE ESTUDIO

2.1. DIAGNÓSTICO SOBRE LAS DIFERENTES ALTERNATIVAS EXISTENTES PARA EL DESARROLLO DE APLICACIONES MÓVILES.

2.1.1 Estudio y evolución de los Sistemas Operativos Móviles

En la actualidad la tecnología avanza a pasos gigantes y en materia de comunicaciones aún más, es asi como los Sistemas Operativos para teléfonos móviles se vuelven cada día más importantes, el uso de la telefonía móvil se convierte en una parte indispensable del accionar diario, es por ello que ha sido importante diseñar sistemas que soporten las aplicaciones demandadas por la sociedad, que sean fluidas, fáciles, accesibles y hasta divertidas.

Un sistema operativo móvil es un programa que se encarga de manejar los procesos básicos de un dispositivo permitiendo el uso de sus diferentes recursos de forma mas simple y orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las diferentes maneras de introducir información en ellos, inicialmente fue desarrollado para computadoras pero actualmente son utilizados en telefónos celulares para tener esa misma interlocución entre el hardware y los programas que el usuario desea utilizar.

Es por eso que las compañías móviles han desarrollado una competencia bastante reñida en cuanto al desarrollo de SO se refiere, según datos de la Superintendencia de Telecomunicación la evolución de la telefonía móvil en el Ecuador inicia en 1983 donde Motorola desarrolló el DinaTAC 8000x que mucho ha cambiado el panorama de las comunicaciones móviles hasta estos días, un hito importante en dicha evolución tecnológica fue la digitalización de las comunicaciones con la llegada a principios de los 90, de la segunda generación de telefonía móvil o 2G, que supuso la integración de servicios de datos junto con los de voz. En 2G se implementaron tecnologías que permitían la conexión móvil a redes de datos, aunque a velocidades lentas y precios elevados. GPRS (Global Packet Radio Service) y a su evolución EDGE (Enhanced Data rates for GSM

Evolution), gérmen del escenario actual en las comunicaciones móviles. Así como a los servicios que se desarrollaron paralelamente: WAP (Wireless Aplication Protocol), que permitía el acceso móvil a Internet; SMS (Short Message Service), que tanta aceptación tuvo hace una década y actualmente en desuso a favor de las redes sociales; o MMS (Multimedia Messaging System), que integra imágenes, audio o video en los SMS.Salvo los SMS, estos servicios apenas llegaron a popularizarse, y se vio la necesidad, más aún tras el vertiginoso avance de las conexiones ADSL en los hogares, que dejó muy altas las expectativas de los consumidores, de mejorar las comunicaciones móviles para ofrecer un servicio comparable al acceso a Internet fijo.

No fue, sin embargo, hasta la segunda mitad de la década del 2000 cuando la migración hacia 3G de las redes móviles permitió a los operadores de telecomunicaciones móviles ofrecer servicios de datos con tasas aceptables a precios razonables.Los estándares para comunicaciones móviles actuales, basados en la tecnología HSDPA (High Speed Downlink Packet Access), que permite descargar datos a una velocidad de hasta 14Mbps, suficiente para soportar las actuales aplicaciones en red y el tráfico multimedia, suponen un paso intermedio hacia la cuarta generación o 4G.

Al tiempo que los operadores de telecomunicaciones, conscientes de la oportunidad de mercado que supone la provisión del servicio de acceso a Internet móvil, han lanzado planes de precios competitivos respecto a las redes de acceso fijo, combinando voz y datos bajo una pseudo-tarifa plana limitada en minutos y Kbytes de descarga.Esta limitación, junto con el abaratamiento de costes perseguido y la tendencia alcista en lo que a tráfico demandado se refiere, ha motivado el desarrollo de nuevos estándares, que, a punto de ser implantados, supondrán la completa evolución hacia la cuarta generación de las comunicaciones móviles o 4G, cuya arquitectura, basada en el protocolo nativo de Internet, IP (Internet Protocol) junto a LTE (Long Term Evolution) la tecnología de acceso radio para datos,pretende dar respuesta a las exigencias del mercado.

En 1996, Palm lanzó el primer sistema operativo móvil, el Palm OS 1.0, que integraba aplicaciones de RIM como correo, agenda, memo pad y tareas. Posteriormente en el año

2000 Microsoft lanzó el Pocket PC 2000 (WinCE 3.0) y un año después, este sistema operativo ya soportaba Messenger y Media Player 8 Enhanced UI. En el año 2003 se lanzó Windows Mobile con bluetooth e internert Explorer.

A finales del año 2000 surgió Symbian, producto de la alianza de varias empresas de telefonía móvil (Nokia, Sony Ericsson, Psion, Samsung, Siemens, Arima, Benq, Fujitsu, Lenovo, LG, Motorola, Mitsubishi Electric, Panasonic, Sharp, etc.) con el objetivo de competir con los S.O. de Palm o el Windows Mobile de Microsoft y ahora Android de Google, iOS de Apple y Blackberry OS de RIM.

Hasta junio de 2007 no vería la luz el S.O. de Apple, el iPhone OS (que en 2010 pasaría a ser iOS), coincidiendo con la salida inicial del iPhone. En el año 2008, Google lanzó la primera versión de su Android. En junio de 2009, se lanza el HP webOS, desarrollado por Palm, ahora propiedad de Hewlett-Packard.

Como se muestra a continuación, la variedad y la cantidad de sistemas operativos que se han desarrollado a lo largo de los últimos años ha significado el esfuerzo de las distintas empresas por posicionarse en el mercado.

Windows Phone, es un sistema operativo móvil desarrollado por Microsoft, a diferencia de su predecesor está enfocado en el mercado de consumo en lugar de en el mercado empresarial; con Windows Phone Microsoft ofrece una nueva interfaz de usuario que integra varios de sus servicios propios como OneDrive, Skype y Xbox Live en el sistema operativo, compite directamente contra Android de Google e iOS de Apple. Su última versión disponible y definitiva es Windows Phone 8.1, lanzado en el 2014, es ya para el año 2015 que se cuenta con el Windows 10 Mobile, disponible para todo tipo de plataformas (teléfonos inteligentes, tabletas y computadoras).

IOS, es un sistema operativo móvil de la multinacional Apple Inc. Originalmente desarrollado para el iPhone (iPhone OS), después se ha usado en dispositivos como el iPod touch y el iPad. No permite la instalación de iOS en hardware de terceros. Los elementos de control consisten de deslizadores, interruptores y botones. La respuesta a las órdenes

del usuario es inmediata y provee una interfaz fluida. La interacción con el sistema operativo incluye gestos como deslices, toques, pellizcos, los cuales tienen definiciones diferentes dependiendo del contexto de la interfaz. Se utilizan acelerómetros internos para hacer que algunas aplicaciones respondan a sacudir el dispositivo (por ejemplo, para el comando deshacer) o rotarlo en tres dimensiones (un resultado común es cambiar de modo vertical al apaisado u horizontal).

iOS cuenta con cuatro capas de abstracción: la capa del núcleo del sistema operativo, la capa de "Servicios Principales", la capa de "Medios" y la capa de "Cocoa Touch". Las versiones actuales del sistema operativo son iOS 8.4.1. y iOS 9, la última.

Es así como nace ***Android***, un sistema operativo y una plataforma de software, basado en Linux para teléfonos móviles. Además de estos, también usan este sistema operativo (aunque no es muy habitual), tablets, netbooks, reproductores de música e incluso PC's. Android permite programar en un entorno de trabajo (framework) de Java, aplicaciones sobre una máquina virtual Dalvik (una variación de la máquina de Java con compilación en tiempo de ejecución).

2.1.2 Marco comparativo entre Sistemas Operativosmóviles investigados

A continuación, se muestra en la tabla 1.un análisis comparativo sobre los 3 sistemas operativos móviles en mención con su análisis respectivo de las plataformas y servicios:

Tabla 1. Diferencias de sistemas operativos

Sistema Operativo	Android	iOS	Windows Phone
Interfaz			
Kernel	Linux	OSx	Windows NT
Tipo de S. Operativo	Abierto	Cerrado	Cerrado
Lenguaje de Programación Nativo	Java	Objective C	C#
Costos por publicación	25 usd una sola vez para publigar en Google Play Store	99 usd anuales + Mac	99 usd anual
Adaptabilidad	Excelente	Excelente	Excelente
Edad media de la Plataforma	Madura	Madura	Joven
Multitarea	SI	SI	SI
Standares Soportados	GSM, CDMA	GSM, CDMA	GSM, CDMA
Hardware soportado	amplia gama de dispositivos	Equipos de plataforma IOS	Limitada gama de dispositivos
Actualización	SI	SI	SI
Cortar/Copiar/Pegar	SI	SI	SI

Programa de productividad	Google Docs	iWork	Office Mobile
Tienda de Libros	Google Books	iBooks	N/A
Tienda de Software	Google Play	App store	Marketplace
Sincronización Wi-Fi	No por defecto	SI	SI
Red Social Integrada	Facebook, Twitter	Facebook, Twitter	Facebook,Twitter, Windows Live
Apps	500000+	650000+	100000+
Soporte para Tablet	SI	SI	NO
Navegador	Basado en Chrome	Safari	Internet Explorer
Mapas	Google Maps	Mapas Apple	Bing Maps
Motor de búsqueda predeterminado	Google	Google	Bing
Expansión de almacenamiento	Micro SD	No	No
Soporte en la Nube	Google Sync, Google Drive	iCloud	SkyDrive
Asistente de Voz	S-Voice (Galaxy S III)	Siri	Tellme
Pantalla de Inicio	Iconos y widgets	Iconos	Baldosas(Tiles)
Interfaz de Usuario	Más Técnico	Fácil	Fácil
Personalización	Profunda	Limitada	Ninguna
Notificaciones	Pull-down	Pull-down	Toast y Banners

Elaborado por: El Autor

2.1.3 Aplicaciónes Móviles

Una aplicación móvil, api o app (en inglés) es una aplicación informática diseñada para ser ejecutada en teléfonos inteligentes, tabletas y otros dispositivos móviles. Por lo general se encuentran disponibles a través de plataformas de distribución, operadas por las compañías propietarias de los sistemas operativos móviles como Android, iOS, BlackBerry OS, Windows Phone, entre otros. Existen aplicaciones móviles gratuitas u otras de pago, donde en promedio el 20-30% del costo de la aplicación se destina al distribuidor y el resto es para el desarrollador. El término app se volvió popular rápidamente, tanto que en 2010 fue listada como Word of the Year (Palabra del Año) por la American Dialect Society.

2.1.3.1 Aplicación Nativa

Una aplicación nativa es la que se desarrolla de forma específica para un determinado sistema operativo, llamado *Software Development Kit* o SDK. Cada una de las plataformas, Adroid, iOS o Windows Phone, tienen un sistema diferente, por lo que si se quiere que la app esté disponible en todas las plataformas se deberán de crear varias apps con el lenguaje del sistema operativo seleccionado.

El desarrollo móvil casi siempre se refiere a aplicaciones nativas. La principal ventaja es la posibilidad de acceder a todas las características del hardware del móvil: cámara, GPS, agenda, dispositivos de almacenimiento y otras muchas. Esto hace que la experiencia del usuario sea mucho más positiva que con otro tipo de apps, además las aplicaciones nativas no necesitan conexión a internet para que funcionen.

La descarga e instalación de estas apps se realiza siempre a través de las tiendas de aplicaciones (app store de los fabricantes). Esto facilita el proceso de marketing y promoción que es vital para dar visibilidad a una app.

Actualmente, entre los principales sistemas operativos iOS, Android y Windows Phone, así como sus mercados App Store, Google Play y Windows Marketplace suman alrededor de 3 millones de aplicaciones disponibles. Este “boom” ha despertado el interés de muchas

personas por desarrollar nuevas aplicaciones que llamen la atención de los consumidores y así lograr hacerse un hueco en el mercado digital.

Estás aplicaciones que a simple vista o por su buen diseño gráfico pueden parecer iguales, se pueden categorizar según cómo han sido desarrolladas, algo que marca sus prestaciones tanto a nivel de respuesta, rapidez o usabilidad. Por tanto, se puede distinguir entre aplicaciones nativas e híbridas, siendo las nativas aquellas que se programan teniendo en cuenta las particularidades de cada plataforma y siendo por lo tanto las que ofrecen mejores prestaciones y las híbridas aquellas que aprovechan un desarrollo común que luego se personaliza para cada tipo de dispositivo: iPhone, Android…etc.

Las aplicaciones nativas se denominan así porque se desarrollan en el lenguaje nativo del propio terminal. Dependiendo de la plataforma para la que se quiera la aplicación, se desarrollará en un lenguaje específico para la misma. Tal como se observa en la Tabla 2.

Tabla 2. Principales Sistemas Operativos

Sistema Operativo	Fabricante	Lenguaje de programación
Android	Google	Java
iOS	Apple	Objective C, Swift
Windows Phone	Microsoft	C#,Visual Basic, NET

Elaborado por: El Autor

Estas aplicaciones se alimentarán de los recursos del propio smartphone, teniendo acceso a diferentes características como la cámara, el GPS, entre otras. Además de esto, tienen muchas ventajas ya que, al estar diseñadas directamente para el software del terminal, tendrán un rendimiento optimizado, así como una interfaz mucho más adaptada al sistema operativo al cual el usuario está acostumbrado. Es por esto que son las favoritas del

mercado debido a que ofrecen resultados más potentes en cuanto a diseño, usabilidad y eficiencia se refiere. Su distribución se hace a través de los marketplaces oficiales de cada sistema operativo, lo que garantiza una visibilidad y seguridad plena.
Otra de sus ventajas es que permiten su uso sin necesidad de conexión a internet, aunque esto no quita que en alguna de sus partes la requiera. Las notificaciones push son otro de su fuerte, así como la creación de un acceso directo en tu pantalla principal después de su instalación.

En conclusión, Android e IOS son los sistemas operativos para Smartphone que más se utilizan en el mercado con mejor soporte y se han integrado a varios servicios web que se ofertan, dejando atrás a Windows Phone que no ha podido aún calar en los clientes de telefonía móvil.

2.2. ESTUDIO DE DESARROLLO DE ANDROID

2.2.1. Definición y Utilidad de Android

Android, es una solución completa de software de código libre basado en el núcleo de Linux creado para una amplia gama de dispositivos con diferentes factores de forma, es un paquete que engloba un sistema operativo, un "Runtime" de ejecución basado en Java, un conjunto de librerías de bajo y medio nivel y un conjunto inicial de aplicaciones destinadas al usuario final. Android se distribuye bajo una licencia libre que permite la integración con soluciones de código propietario.

Los propósitos principales de Android son la creación de una plataforma de software abierto disponible para los operadores, OEMs y desarrolladores para hacer que sus ideas innovadoras en una realidad y para introducir un producto de éxito, en el mundo real que mejora la experiencia móvil de los usuarios.

La utilización del sistema operativo Android es muy importante debido a que es un sistema de código abierto, que:

- Dispone de un sitio web propio, Google Play, para la descarga de aplicaciones.
- La adaptabilidad de Android al usuario es más versátil.
- Se integra a los servicios de Google.
- El compartir contenidos es más dinámico en Android que en IOS.
- El costo de las aplicaciones.
- La variedad de modelos y marcas de preferencia del usuario están a disposición en Android.
- Es el sistema operativo utilizado por varios fabricantes de teléfonos móviles y tabletas, como: Samsung, LG, HTC, Motorola, Sony, Micromax y otros. Por lo tanto, el usuario cuenta con amplias posibilidades de elección y cada una de estas marcas cuenta con dispositivos de diferente gama en prestaciones y costo.
- Se encuentra disponible en Internet gran cantidad de información, tutoriales, cursos, ejemplos de aplicaciones y libros electrónicos sobre el sistema operativo Android.
- Dispone de diferentes versiones en las cuales se pueden desarrollar varias aplicaciones como se observa en la tabla 3.

En conclusión, el sistema operativo Android es más ventajoso tanto para el desarrollo de aplicaciones como para los usuarios debido a que por tratarse de un sistema de código abierto está libre de licencias, no se realizá pago alguno por distribución o desarrollo, se encuentra a disposición una gran cantidad de información en Internet y es el más utilizado en los dispositivos móviles de los fabricantes más reconocidos a nivel mundial.

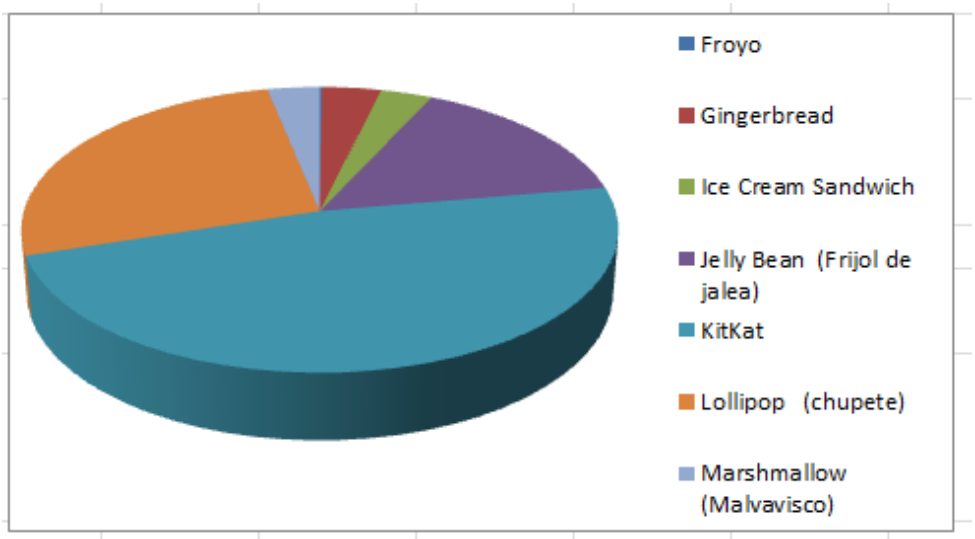
Tabla 3. Versiones de Android

Versión	Nombre de código	API	Distribución
2.2	Froyo	8	0.1%
2.3.3 -2.3.7	Gingerbread (Pan de jengibre)	10	2.6%
4.0.3 -4.0.4	Ice Cream Sandwich	15	2.3%

4.1.x	Jelly Bean (Frijol de jalea)	16	8.1%
4.2.x	Jelly Bean (Frijol de jalea)	17	11.0%
4.3	Jelly Bean (Frijol de jalea)	18	3.2%
4.4	KitKat	19	34.3%
5.0	Lollipop (chupete)	21	16.9%
5.1	Lollipop (chupete)	22	19.2%
6.0	Marshmallow (Malvavisco)	23	2.3%

Elaborado por: El Autor

Ilustración 2. Versiones de plataforma Android



Elaborado por: El Autor

2.2.2. Características generales de Andorid

Android surge como resultado de la "Open Handset Alliance" un consorcio de 48 empresas distribuidas por todo el mundo con intereses diversos en la telefonía móvil y un compromiso de comercializar dispositivos móviles con este sistema operativo. El desarrollo viene avalado principalmente por Google (tras la compra de Android Inc. en 2005).

Android es una plataforma de código abierto, lo que significa que no solamente pueden mejorarla los desarrolladores de Google, sino que también se nutre de las aportaciones de desarrolladores externos. Originalmente, el objetivo de implementación para Android fue el área de teléfonos móviles, incluso teléfonos inteligentes. Sin embargo, el rango completo de servicio de computación de Android y el amplio soporte funcional tienen el potencial para extenderse más allá del mercado de teléfonos móviles. Android puede ser útil para otras plataformas y aplicaciones.

La diferencia de Android con otros sistemas operativos es que gracias a su entorno de desarrollo, se encuentra en la interfaz de programación de aplicaciones (API) que pone a disposición de los desarrolladores novatos la oportunidad que cualquier persona que sepa programar puede crear nuevas aplicaciones, widgets, o incluso, modificar el propio sistema operativo, dado que Android es de código libre, por lo que sabiendo programar en lenguaje Java, va a ser muy fácil comenzar a programar en esta plataforma.

Android posee un conjunto de aplicaciones nativas como:

- Un cliente de correo electrónico.
- Una aplicación de gestión de SMS.
- Un conjunto completo para la administración de la información personal que incluye una lista de contactos y un calendario.
- Un navegador web basado en WebKit.
- Un reproductor de música y galería de fotos.
- Una aplicación de la cámara y grabación de vídeo.
- Una calculadora.
- Pantalla de inicio.
- Un reloj con alarma.

A continuación, se detallan las principales características de Android:

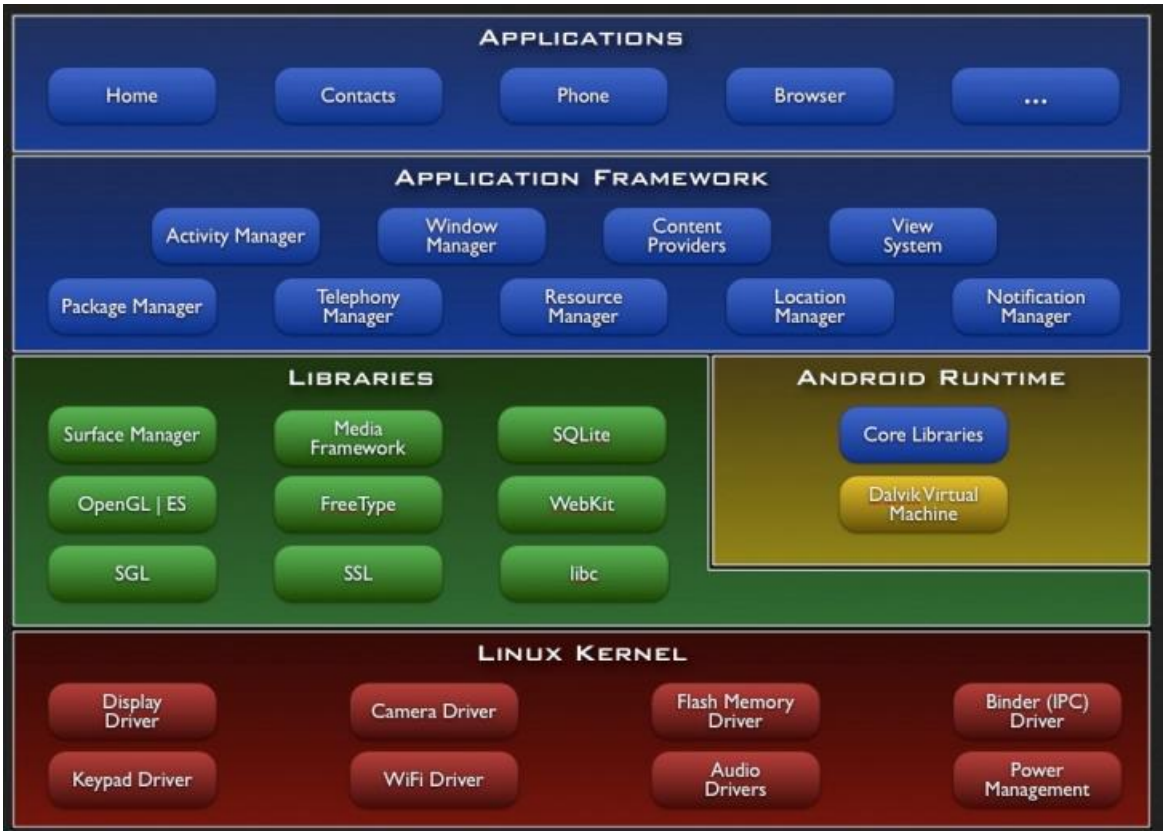
- Núcleo basado en Linux y de código libre. Se puede usar y adecuar el sistema sin pagar algún tipo de licencia.
- Busca el desarrollo rápido de aplicaciones, que sean reutilizables y verdaderamente portables entre diferentes dispositivos.
- Los componentes básicos de las aplicaciones se pueden sustituir fácilmente por otros.
- Cuenta con su propia máquina virtual, Dalvik, que interpreta y ejecuta código escrito en Java.
- Permite la representación de gráficos 2D y 3D.
- Posibilita el uso de bases de datos para almacenamiento estructurado que se integra directamente con las aplicaciones.
- Soporta un elevado número de formatos multimedia, audio y vídeo: WebM, H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF y BMP.
- Servicio de localización GSM.
- Controla los diferentes elementos hardware: Bluetooth, Wi-Fi, cámara fotográfica o de vídeo, GPS, acelerómetro, infrarrojos, etc., siempre y cuando eldispositivo móvil lo contemple.
- Navegador integrado basado en el motor open Source Webkit.
- Telefonía GSM dependiente del terminal.
- Bluetooth, EDGE, 3g y Wifi dependiente del terminal.
- Entorno de desarrollo (emulador, herramientas de depuración, perfiles de memoria y funcionamiento, plugin para Eclipse IDE).
- Las interfaces se hacen en formato xml, lo que permite el uso de una misma interfaz en dispo sitivos de distintos tamaños de pantallas.
- Nivel de seguridad: Los programas se encuentran separados unos de otros. Cada aplicación dispone distintos tipos de permisos que limitan su ango de actuación.

2.2.3. Arquitectura de Android

La arquitectura de Android está formada por varios niveles o capas lo que facilita el desarrollo de aplicaciones ya que permite trabajar con las capas inferiores por medio de las librerías evitando programar a bajo nivel y lograr que los componentes de hardware del dispositivo móvil interactúen con la aplicación

La arquitectura interna de la plataforma Android, está básicamente formada por 4 componentes tal como se muestra en la Ilustración 3.

Ilustración 3.Estructura de Android (Tomado de [2])



2.2.3.1 Aplicaciones

Todas las aplicaciones creadas con la plataforma Android, incluirán como base un cliente de email (correo electrónico), calendario, programa de SMS, mapas, navegador, contactos, y algunos otros servicios mínimos. Todas ellas escritas en el lenguaje de programación Java.

2.2.3.2 Framework de aplicaciones

Todos los desarrolladores de aplicaciones Android, tienen acceso total al código fuente usado en las aplicaciones base. Esto ha sido diseñado de esta forma, para que no se generen cientos de componentes de aplicaciones distintas, que respondan a la misma acción, dando la posibilidad de que los programas sean modificados o reemplazados por cualquier usuario sin tener que empezar a programar sus aplicaciones desde el principio, el framework esta conformado por:

- **Activity Manager:** Se encarga de administrar la pila de actividades de la aplicación, así como su ciclo de vida.
- **Windows Manager:** Se encarga de organizar lo que se mostrará en pantalla.
- **Content Provider:** Esta librería crea una capa que encapsula los datos que se compartirán entre aplicaciones para tener control sobre cómo se accede a la información.
- **Views:** Las vistas son elementos que ayudan a construir las interfaces de usuario: botones, cuadros de texto, listas y hasta elementos más avanzados como un navegador web o un visor de Google Maps.
- **Package Manager:** Esta biblioteca permite obtener información sobre los paquetes instalados en el dispositivo Android, además de gestionar la instalación de nuevos paquetes.
- **Location Manager:** Permite determinar la posición geográfica del dispositivo Android mediante GPS o redes disponibles y trabajar con mapas.
- **Sensor Manager:** Permite manipular los elementos de hardware del dispositivo móvil como el acelerómetro, giroscopio, brújula, etc.

- **Cámara:** Con esta librería se puede hacer uso de la(s) cámara(s) del dispositivo para tomar fotografías o para grabar vídeo.
- **Multimedia:** Permite reproducir y visualizar audio, vídeo e imágenes en el dispositivo.

2.2.3.3 Librerías

Android incluye en su base de datos un set de librerías C/C++, que son expuestas a todos los desarrolladores a través del framework de las aplicaciones Android System C library, librerías de medios, librerías de gráficos, 3D, SQlite, etc.

2.2.3.4 Runtime de Android

Android incorpora un set de librerías que aportan la mayor parte de las funcionalidades disponibles en las librerías base del lenguaje de programación Java. La Máquina Virtual está basada en registros, y corre clases compiladas por el compilador de Java que anteriormente han sido transformadas al formato. dex (Dalvik Executable) por la herramienta ''dx''.

2.2.3.1 Kernelde Linux

Es la base de Android, utilizado por su robustez demostrada y por la implementación de funciones básicas para cualquier sistema operativo, como: seguridad, administración de memoria y procesos, implementación de conectividad de red y varios controladores para la comunicación con los dispositivos físicos.

Android emplea como base el kernel de Linux, pero los dos sistemas no son lo mismo, Android no cuenta con un sistema nativo de ventanas de Linux, ni tiene soporte para la librería estándar de C, ni tampoco es posible utilizar la mayoría de aplicaciones de GNU de Linux. Además, sobre todo lo implementado en el kernel de Linux, Android agrega algunas cosas específicas para plataformas móviles como la comunicación entre procesos, la forma de manejar la memoria compartida y la administración de energía.

2.2.4. Componentes de una aplicación Android

Toda aplicación de Android esta basada en ciertos componentes básicos con los que se contruye una aplicación, misma que será el producto de la combinación de una o mas de dichos componentes y se deben declarar en el fichero AndroidManifiest.xml en el que cual se definen todos los componentes de la aplicación, así como los permisos requeridos, resursos y librerías utilizados.

La estructura de programación que se usa para activar cada uno de estos componentes en una aplicación, se la puede encontrar detallada en la página oficial de desarrolladores Android en donde además existen varios ejemplos para comprender su funcionamiento, el objetivo de la presente es brindar una idea de cómo trabajan las aplicaciones en este sistema operativo. Existen cuatro tipos de componentes básicos, cada uno de los cuales sirve para un propósito específico y poseen un ciclo de vida propio que define cuando es creado y destruido.

Ilustración 4. Componentes de una aplicación (Tomado de [3])

2.2.4.1 Activities

Es una pantalla que muestra una interfaz de usuario, dependiendo de la aplicación, por ejemplo, si se trata de una aplicación cliente de Twitter, una Activity puede ser aquella que muestra el Timeline del usuario. De esta manera una aplicación puede estar compuesta de varias activities que en conjunto forman la aplicación.

2.2.4.2 Servicios

Es un componente que corre en segundo plano y realiza operaciones que toman un tiempo considerable. Los servicios no proveen una interfaz para el usuario, por ejemplo, un servicio puede reproducir música mientras el usuario se encuentra en una aplicación diferente.

2.2.4.3 Proveedor de contenidos (Content Provider)

Administra un conjunto compartido de datos de una aplicación, se puede almacenar datos en el sistema de archivos, una base de datos o la web, cualquier lugar al que la aplicación tenga acceso. A través de un proveedor de contenidos otras aplicaciones pueden consultar o incluso modificar los datos en caso de que sea permitido. Un ejemplo de un proveedor de contenidos es aquel que administra la información de los contactos almacenados en el dispositivo.

2.2.4.4 Broadcast receivers

Es un componente que responde a los anuncios del sistema como por ejemplo cuando la pantalla se apaga, existe poca carga en la batería, etc. No poseen una interfaz de usuario como las Activities, pero usan notificaciones que permite alertar al usuario cuando un evento de estas características ha ocurrido.

2.2.5 Recursos de una aplicación Android

Las aplicaciones desarrolladas para Android además del código fuente, están compuestas de otros recursos como imágenes, archivos de audio o cualquier otro elemento que forme parte de la interfaz del usuario. Para definir menús, estilos, colores, animaciones, etc. Se

hace uso de archivos XML debido a que permiten modificar o actualizar características de la App sin necesidad de modificar el código del programa. El uso de los recursos por una aplicación hace que sea fácil actualizar varias características sin modificar el código y proporcionan un conjunto de alternativas, los recursos permiten al desarrollador optimizar la aplicación para una variedad de configuraciones de dispositivos, como diferentes lenguajes y tamaños de pantalla. Cada recurso que sea incluido en un proyecto de desarrollo de una aplicación Android, tendrá su propio identificador ID, definido por las herramientas de desarrollo SDK el cual es usado para hacer referencia al mismo desde el código de la App o desde otros recursos definidos en XML. Una de las propiedades más importantes del suministro de recursos por separado a partir del código fuente, es la capacidad para que el programador pueda proporcionar recursos alternativos a diferentes configuraciones de dispositivos.

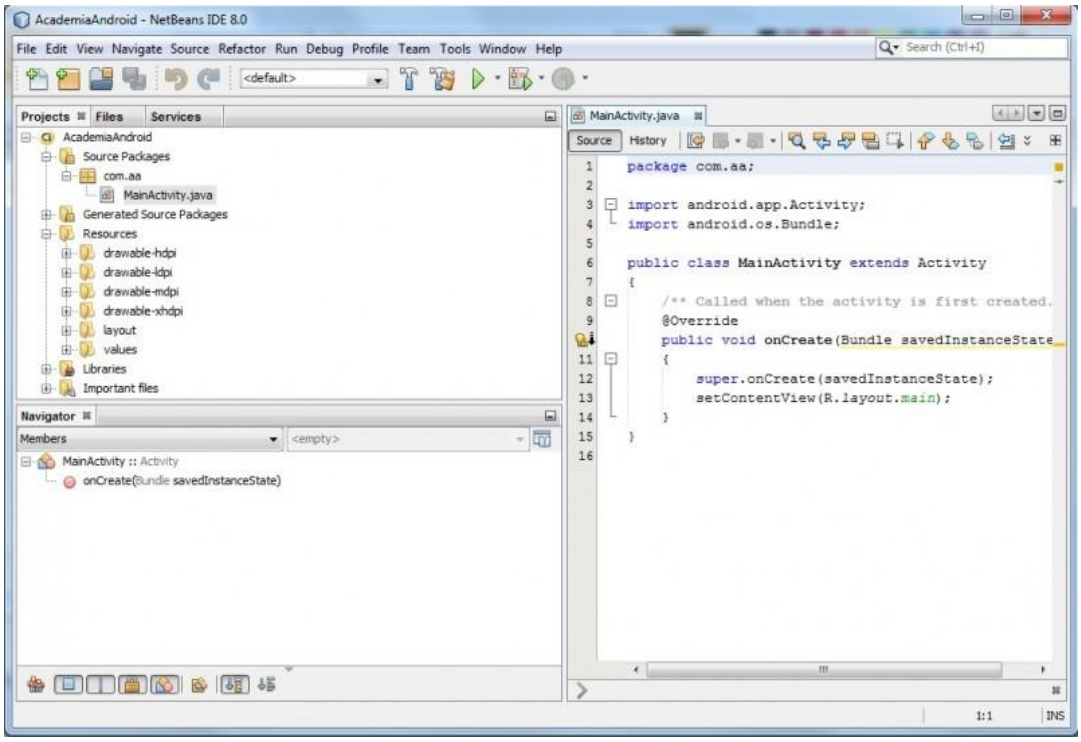
2.2.6 Análisis de los IDE de desarrollo para plataforma Android

Android es el sistema operativo móvil con más popularidad y que sigue creciendo a pasos agigantados. Es por ello que se realiza el siguiente análisis para conocer qué herramientas existen para poder desarrollar una aplicación de forma nativa para el sistema operativo Android.

El lenguaje que Google pensó que debería de usarse para programar aplicaciones para Android es Java. A continuación, se realiza un análisis a los distintos entornos de desarrollo (IDE) disponibles y más populares por los cuales empezar a desarrollar aplicaciones para Android.

Un IDE es un programa informático compuesto por un conjunto de herramientas de programación. Es un entorno de programación empaquetado como un programa o aplicación, que provee de un marco de trabajo agradable para la mayoría de los lenguajes de programación. Constan entre sus características básicas con: Editor de código, compilador, depurador (debbuger), constructor de interfaz gráfica, tal como se muestra en la Ilustracón 5.

Ilustración 5. Características de un IDE (Tomado de [4])



Entre los IDE más populares para el desarrollo de aplicaciones Android se destacan los siguientes:

- Eclipse
- IntelliJ
- Android Studio
- Aide

2.2.6.1 Eclipse

Eun entorno de desarrollo de código abierto y gratuito, cuyo diseño sigue un patrón de actualización basado en plugins. Su objetivo es convertirse en una plataforma de integración de herramientas de desarrollo. Es un IDE que se podría denominar genérico, ya que no fue concebido para ser utilizado con un solo lenguaje de programación, sino que es compatible con una gran variedad de lenguajes, sus principales características se las puede observar en la Tabla 4.

Tabla 4. Características del entorno de desarrollo en eclipse.

<table>
<tr><th>Gestión de Proyectos</th><th>Depurador de Código</th><th>Perspectivas, Editores y Vistas</th><th>Colección de Plugins</th></tr>
<tr><td rowspan="2">El desarrollo sobre Eclipse se basa en proyectos, como pueden ser el código fuente, documentación, ficheros, etc..</td><td>Incluye un potente depurador de código, fácil y intuitivo, que proporciona de forma gráfica una opción de mejorar los proyectos.</td><td rowspan="2">El concepto de trabajo se basa en las perspectivas, que son una preconfiguración de ventanas y editores que permiten trabajar en un determinado entorno de trabajo de forma óptima.</td><td>Están disponibles una gran cantidad de plugins, tanto desarrollados por Eclipse como de terceros.</td></tr>
<tr><td>Dispone de una perspectiva dedicada a la depuración donde se puede realizar y supervisar dicha tarea.</td><td>actualmente el número de ellos es muy alto, rondando los 1.280 plugins que pueden aumentar las funcionalidades del IDE</td></tr>
</table>

2.2.6.2 IntelliJ IDEA

Es un Ambiente de desarrollo integrado (IDE) para el desarrollo de programas informáticos. Es desarrollado por JetBrains (Anteriormente conocido como IntelliJ), y está disponible en dos ediciones: community edition, y edición comercial. IntelliJ IDEA no está basada en Eclipse como MyEclipse o Oracle Enterprise Pack para Eclipse.

“La primera versión de IntelliJ la IDEA fue liberada en enero 2001, y en ese tiempo fue uno de los primeros Java IDE disponibles con avanzada navegación de código y capacidades de refactorización de código integrado.

En el 2010 Infoworld, IntelliJ recibió la puntuación de centro de prueba más alta fuera de las cuatro herramientas de programación superiores de Java: Eclipse, IntelliJ IDEA, NetBeans y Oracle JDeveloper.

En diciembre 2014, Google anunció versión 1.0 de Android Studio, un código abierto IDE para aplicaciones de Android, basados en el código abierto en la edición comunitaria de IntelliJ IDEA. Otros entornos de desarrollo se basaron en IntelliJ incluidos AppCode, PhpStorm, PyCharm, RubyMine, WebStorm, y MPS;" [4] las características más importantes que incorpora IntelliJ Community Edition se encuentran en la Tabla 5.

Tabla 5. Características de IntelliJ Community Edition

Análisis de Código	Potente Editor	Herramientas integradas de Android	Aumento de Productividad	Lenguajes Soportados
El editor resalta advertencias y errores inmediatamente permitiendo aplicar una solución más rápida.	Permite finalización de código inteligente y autocompletado más sofisticado y sensible.	Posee un diseñador de interfaz de usuario que permite arrastrar y soltar elementos, apoyo a distintos diseños y tipos de pantalla.	Añade soporte para Maven y Gradle. Posee herramientas integradas para pruebas unitarias y de cobertura.	Soporta varios lenguajes basados en JVM populares como son: Java, Scala, Groovy, Clojure y Kotin.
Permite programación con intenciones y soluciones rápidas, apoyo a los controles Android Lint, perfiles de inspección compartidos.	Recursos de previsualización y refactorizaciones avanzadas y seguras.	Permite integración con Logcat. Posee filtros de búsqueda personalizados.	Posee un control de versiones compatible con Gib, GibHub, SVN entre otros.	

La versión de pago contiene varias características como un mayor soporte a lenguajes, como se señala a continuación:

- PHP, Python y Rubí.
- SQL, incluyendo PostgreSQL, MySQL, Oracle, SQL Server, etc…Además añade soporte avanzado para los marcos y estándares web más importantes.
- Desarrollo fácilmente con Spring MVC, Webflow, Jugar, Grails, Servicios Web, JSF, Struts, Flex y otros marcos.

- Incluye asistencia de código final para HTML5, CSS3, SASS, MENOS, Javascript, CoffeScript, Node js, ActionScript y otros lenguajes.

2.2.6.3 AIDE

Se trata de un IDE para Android que permite programar y compilar los proyectos directamente en el dispositivo Android. Entre sus características principales ofrece la posibilidad de seguir programando los proyectos directamente en tablets y poder emular directamente en el dispositivo la aplicación.

AIDE permite de una forma muy sencilla programar en Android por medio de Java aplicaciones como Eclipse, Android Studio, en el cuál las características principales son:

- Crear una aplicación de muestra con un solo clic
- Construir aplicaciones Java / XML. Construir aplicaciones C / C++.
- Ejecutar las aplicaciones con un solo clic. Compilación incremental para ahorrar tiempo.
- Utiliza classpath para la compatibilidad con Eclipse.
- Abrir los proyectos del nuevo Android Studio.
- Visor LogCat integrado.
- Visor de errores en tiempo real.
- Completo autocompletar.

2.2.6.4 Android Studio

Es un entorno de desarrollo integrado para la plataforma Android, este IDE reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android, está basado en el software IntelliJ IDEA de JetBrains, y es publicado de forma gratuita a través de la Licencia Apache 2.0. Está disponible para las plataformas Microsoft Windows, Mac OS X y GNU/Linux, sus características principales son:

- Renderización en tiempo real.
- Consola de desarrollador: consejos de optimización, ayuda para la traducción, estadísticas de uso.
- Soporte para construcción basada en Gradle.
- Refactorización específica de Android y arreglos rápidos.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versiones, y otros problemas.
- Plantillas para crear diseños comunes de Android y otros componentes.
- Soporte para programar aplicaciones para Android Wear.

Para realizar un análisis sobre la mejor IDE de desarrollo Android se tomarán en cuenta los siguientes parámetros para evaluar:

- Experiencia de uso (que experiencia tengo manejando el IDE)
- Recomendación de Fabricante (android studio recomendo porGoogle)
- Facilidad de manejo curva de aprendizaje
- Costo de licenciamiento

2.2.7 Elección del IDE de desarrollo

La elección del IDE de desarrollo se debe realizar de acuerdo a: la información que se pueda obtener en la web, curva de aprendizaje, estándares de desarrollo que se adapten a la aplicación en desarrollo.

Por estos motivos Android Studio es el que más se acerca a las características de desarrollo deseada ya que al pertenecer a google se adapta a los estándares de desarrollo necesitados, es gratuito, se encuentra soporte y foros en donde despeja dudas y cuestionamientos entre otras características.

2.3 ANÁLISIS DE LA APLICACIÓN DE GOOGLE PARA LA GEOLOCALIZACIÓN Y MENSAJERÍA INSTANTÁNEA

2.3.1 Estudio de Geo localización en Google

2.3.1.1 Google Maps

Es un servidor de aplicaciones de mapas en la web, ofrece imágenes de mapas desplazables, acercamientos o alejamientos para mostrar el mapa, así como fotografías por satélite del mundo. El usuario puede controlar el mapa con el ratón o las teclas de dirección para moverse a la ubicación que se desee. Existe una variante a nivel entorno de escritorio llamada Google Earth.

2.3.1.2 Google Maps API

Google Maps fue una de las primeras aplicaciones basadas en AJAX de uso masivo por parte de los usuarios, además ofrece una API con la que de forma gratuita se puede desarrollar aplicaciones a medida basadas en los mapas de Google ademñas de poder integrarlos en otras aplicaciones e incluso hacer "mash-up" o mezclas de Google Maps y otras aplicaciones web que también disponen de una API pública.

Google maps API permite incrustar los mapas de Google Maps en páginas web o sistemas móviles mediante JavaScript y proporciona:

- Utilidades para manipular los mapas
- Añadir contenidos al mapa mediante diversos servicios
- Servicio gratuito para los usuarios Google

Antes de utilizar la API de los mapas de Google, es necesario obtener una clave personal y única para cada sitio web donde se quiere utilizar. “El uso de la API es gratuito para cualquier aplicación que pueda ser accedida libremente por los usuarios. La clave de la API se puede obtener desde: http://www.google.com/apis/maps/

Para usos comerciales de la API también existen servicios de pago que requieren el uso de otras claves.

Las claves se solicitan por cada ruta del servidor. De esta forma, si se solicita una clave para http://www.misitio.com/ruta1, cualquier aplicación o página que se encuentre bajo esa ruta del servidor podrá hacer uso de la API de los mapas con esa clave.

Si no se dispone de un sitio web público, es posible indicar como servidor el valor http://localhost para poder hacer pruebas en un servidor local. Para solicitar la clave, es necesario disponer de una cuenta de usuario de Google (se puede utilizar por ejemplo la cuenta de Gmail). La clave generada depende por tanto del dominio de la aplicación y de la cuenta de usuario.

Las claves de la API de Google Maps consisten en una cadena de texto muy larga con un aspecto similar al siguiente:

ABQIAAAA30JtKUU8se7KKPRGSfCMBT2yXp_ZAY8_ufC3CFXhHIE1NvwkxRZNdns2BwZvEY-V68DvlyUYwi1-Q.

Una vez obtenida la clave, cualquier página que quiera hacer uso de la API debe enlazar el siguiente archivo de JavaScript:

<scriptsrc="http://maps.google.com/maps?file=api&v=2&hl=es&key=ABQIAAAA30JtKUU8se7KKPRGSfCMBT2yXp_ZAY8_ufC3CFXhHIE1NvwkxRZNdns2BwZvEY-V68DvlyUYwi1-Q" type="text/javascript"></script>

Los parámetros necesarios son file, que indica el tipo de archivo que se quiere cargar (en este caso la API), v, que indica la versión de la API (en este caso 2), hl, que permite indicar el idioma en el que se muestran los mapas (si no se indica este parámetro, los mapas se muestran en inglés) y el parámetro key, que contiene la clave que se acaba de obtene". [5]

2.3.1.3 Google Developer Console

Es el cloud de Google donde se almacenan herramientas de desarrollo de software, interfaces de programación de aplicaciones (API), y recursos técnicos. El sitio contiene documentación sobre el uso de herramientas y APIs, incluyendo grupos de discusión y blogs para los desarrolladores que utilizan productos de Google para desarrollo.

Las API´s más importantes son: Google maps, youtube Google apps entre otras, además de contar con varios productos y herramientas construidas específicamente para desarrolladores.

Google App Engine es un servicio de alojamiento de aplicaciones web. Un Hosting que permite el controlar las versiones de un proyecto para los usuarios que desarrollan con código fuente abierto. Google Web Toolkit (GWT) permite a los desarrolladores crear aplicaciones Ajax utilizando lenguajes de programación Java. El sitio también contiene información de referencia para los productos basados en la comunidad de desarrolladores que utilizan Google y están involucrados como Android.

2.3.2 Chat: Plataforma de desarrollo Sinch

Sinch ofrece una plataforma para la comunicación en tiempo real a través de Internet, se compone de diferentes kits de desarrollo de software denominados los SDK Sinch que se integra con el teléfono inteligente o una aplicación web basada en la nube y los servicios de back-end. En conjunto, permiten la comunicación de voz y mensajes de texto instantáneos basado en la aplicación.

El SDK Sinch es un producto que hace que la adición de llamadas de voz y / o mensajería instantánea para aplicaciones móviles fáciles. Se ocupa de toda la complejidad de la señalización y gestión de audio, mientras que le proporciona la libertad para crear una interfaz de usuario impresionante.

Dentro de la aplicación de mensajería instantánea se añade una capa social y una clave para la aplicación para permitir la mejor conexión de sus usuarios entre sí y así generar un

mejor servicio de la aplicación desarrollada. Al igual que WhatsApp, WeChat, y Facebook Messenger, se puede agregar un servicio de mensajería instantánea sin complejidad, pago por uso es decir no hay cargos mensuales adicionales.

2.4 TECNOLOGÍAS DE INTEROPERABILIDAD Y COMUNICACIÓN

2.4.1 Conexiones Móviles

2.4.1.1 Conexión WI-FI

WiFi, permite a los usuarios establecer conexiones a Internet sin ningún tipo de cables y puede encontrarse en cualquier lugar que se haya establecido un "punto caliente" o hotspot WiFi.

Los dispositivos habilitados con wifi (como una computadora personal, un televisor inteligente, una videoconsola, un teléfono inteligente o un reproductor de música) pueden conectarse a internet a través de un punto de acceso de red inalámbrica. Dicho punto de acceso tiene un alcance de unos veinte metros en interiores, distancia que es mayor al aire libre.

El objetivo de WiFi es fomentar las conexiones inalámbricas y facilitar la compatibilidad de los distintos equipos. Todos los productos con conectividad WiFi tienen certificada su interoperabilidad.

Existen diversos tipos de **Wi-Fi**, y son los siguientes:

- "Los estándares IEEE 802.11b e IEEE 802.11g disfrutan de una aceptación internacional debido a que la banda de 2.4 GHz está disponible casi universalmente, con una velocidad de hasta 11 Mbps y 54 Mbps, respectivamente (Dispositivos wireless).
- En la actualidad ya se maneja también el estándar IEEE 802.11a, conocido como WIFI 5, que opera en la banda de 5 GHz para las tecnologías (Bluetooth, microondas,

ZigBee, WUSB). Su enlace es algo menor que el de los estándares que trabajan a 2.4 GHz (un 10% a mayor frecuencia, menor alcance).” [6]

Uno de los principales defectos atribuidos a la conectividad WiFi es su poca seguridad, sin embargo, diversos protocolos de cifrado que permiten codificar la transmisión de los datos y garantizar su confidencialidad.

La infraestructura de una conexión WiFi incluye puntos de acceso (emisores remotos), routers (que reciben la señal que emite el operador de telefonía) y dispositivos de recepción (tarjetas USB, PCI o PCMCIA).

WiFi permite que cualquier persona que tenga una computadora portátil con los componentes necesarios para el acceso a una red inalámbrica pueda ingresar a una gran cantidad de hoteles o restaurantes y conectarse a Internet con su propio equipo.

En los últimos años, se ha visto un crecimiento considerable en la adopción del estándar WiFi por parte de usuarios de todos los niveles, en todas partes del mundo. Es cierto que tiene un gran potencial; sin embargo, como ocurre con cualquier otra tecnología, también acarrea una serie de problemas.

2.4.1.2 Conexión 3G o Redes de Terecera Generación

La tercera generación de redes inalámbricas (3G) es un término que se utiliza para describir a la generación actual de servicios móviles, los cuales proveen una mejor calidad de voz que las redes 2G, Internet de alta velocidad y servicios multimedia.

Las redes 3G deben ser capaces de proporcionar servicios de datos con una transmisión mínima de 144 kbps para ambientes móviles (exteriores) y 2 Mbps para ambientes fijos (interiores). Basado en estos requerimientos la ITU aprobó cinco candidatos como interfaces de radio para los estándares IMT-2000 en 1999 como parte de la recomendación ITU-R M.1457. WiMAX también fue considerado como

parte de las tecnologías 3G, pero esto fue hasta el 2007. En la Tabla 6 se muestran algunos parámetros relacionados con estas tecnologías:

Tabla 6. Estándares 3G/IMT-2000

<table>
<tr><th>Nombre según ITU IMS-2000</th><th colspan="2">Nombre Comercial</th><th>Tecnología para datos</th><th>Pre-4G</th><th>Duplex</th><th>Acceso Múltiple</th><th>Descripción</th><th>Cobertura</th><th>Grupo</th></tr>
<tr><td>TDMA Single</td><td colspan="2" rowspan="3">EDGE (UWT-136)</td><td rowspan="3">EDGE Evolution</td><td rowspan="3">Ninguno</td><td rowspan="6">FDD</td><td rowspan="5">TDMA</td><td></td><td>Mundial</td><td rowspan="3">3GPP</td></tr>
<tr><td>Carrier</td><td>Evolución</td><td>excepto</td></tr>
<tr><td>(IMT-SC)</td><td>GSM/GPRS</td><td>Japón y Corea del Sur</td></tr>
<tr><td>CDMA Multi-Carrier</td><td colspan="2" rowspan="2">CDMA 2000</td><td rowspan="2">EV-DO</td><td rowspan="2">UMB</td><td rowspan="2">Evolución cdmaOne</td><td rowspan="2">América, Asia, otros</td><td rowspan="2">3GPP2</td></tr>
<tr><td>(IMT-MC)</td></tr>
<tr><td>CDMA Direct</td><td rowspan="5">UMTS</td><td>1.WCDMA</td><td rowspan="5">HSPA</td><td rowspan="5">LTE</td><td rowspan="4">CDMA</td><td rowspan="5">Familia de Estándares</td><td></td><td rowspan="5">3GPP</td></tr>
<tr><td>Spread</td><td>2. TD-CDMA</td><td>1.Mundial</td></tr>
<tr><td>(IMT-DS)</td><td rowspan="3">3.TD-SCDMA</td><td>2.Europa</td></tr>
<tr><td>CDMA TDD</td><td rowspan="3">TDD</td><td>3.China</td></tr>
<tr><td>(IMT-TC)</td><td></td></tr>
<tr><td>FDMA/TDMA</td><td colspan="2" rowspan="2">DECT</td><td colspan="2" rowspan="2">Ninguno</td><td>FDMA</td><td>Servicio</td><td>Europa,</td><td rowspan="2"></td></tr>
<tr><td>(IMT-FT)</td><td>TDMA</td><td>Fijo</td><td>USA</td></tr>
<tr><td>IP OFDMA</td><td colspan="2"></td><td colspan="2">WiMAX (IEEE 802.16)</td><td>OFDMA</td><td></td><td>Mundial</td><td>WiMAX</td></tr>
</table>

Elaborado por: El Autor

Las redes 3G se diseñaron tomando en cuenta las siguientes características:

- **Incrementar significativamente la capacidad del sistema de radio y las tasas de datos por usuario que se tenían en los sistemas 2G:** Los sistemas de radio 3G deben soportar tasas de datos hasta de 144 kbps para usuarios móviles en un vehículo, 384 kbps para peatones y hasta 2 Mbps para usuarios estacionarios o fijos.

- **Soporte de servicios de datos, voz y multimedia basados en IP:** El objetivo de diseñar las redes 3G es que no exista una frontera entre la comunicación y las personas que acceden a través de Internet y sus servicios por medio de las redes 3G.
- **Mejoras al soporte de calidad del servicio (QoS):** Los sistemas 3G buscan proveer mayor soporte de calidad del servicio, debido a que estos sistemas proveen servicios de datos, voz y video en tiempo real, flujos de datos, así como descargas de video y audio.
- **Mejorar interoperabilidad:** La interoperabilidad entre los sistemas 3G con los 2G es muy importante para procesos de roaming entre diferentes proveedores de servicio, tecnologías de radio y países.

El ancho de banda y la información disponible en los dispositivos 3G permite aplicaciones que con las generaciones anteriores no eran posibles:

- **TV móvil:** El proveedor de 3G direcciona un canal de TV al teléfono del subscripto Video sobre demanda: El proveedor envía un archivo de video al teléfono del subscriptor
- **Videoconferencia:** Los subscriptores pueden realizar video llamadas.
- **Servicios basados en la localización:** Dependiendo del lugar, el proveedor de servicios permite conocer el estado del clima, del tráfico o bien algún otro tipo de información.

Dos organizaciones a nivel mundial son las encargadas del desarrollo de extensiones, así como evolución de los estándares de redes inalámbricas 3G: 3GPP y 3GPP2. Ya que MBMS es una tecnología perteneciente a los estándares hechos por 3GPP, solamente se realiza el análisis a las redes basadas en conmutación de paquetes IP que define el grupo 3GPP.

2.4.2 GPS funcionamiento e integración con dispositivos móviles

El GPS es un sistema que permite determinar en toda la Tierra la posición, localización, velocidad y altura de un objeto (una persona, un vehículo), las 24 horas del día, bajo cualquier condición atmosférica y en cualquier punto del globo terrestre, con una precisión de hasta centímetros (si se utiliza GPS diferencial), aunque lo habitual son unos pocos

metros de precisión. El sistema fue desarrollado, instalado y empleado por el Departamento de Defensa de los Estados Unidos.

2.4.2.1 Funcionamiento GPS

El funcimaiento del GPS se realiza a través de una red de 32 en total contando los satélites que mejoran la precisión, en órbita sobre el planeta tierra, a 20.200 km de altura, con trayectorias sincronizadas para cubrir toda la superficie de la Tierra. Cuando se desea determinar la posición, el receptor que se utiliza para ello localiza automáticamente como mínimo cuatro satélites de la red, de los que recibe unas señales indicando la identificación y la hora del reloj de cada uno de ellos.

Con base en estas señales, el aparato sincroniza el reloj del GPS y calcula el tiempo que tardan en llegar las señales al equipo, y de tal modo mide la distancia al satélite mediante el método de trilateración inversa, la cual se basa en determinar la distancia de cada satélite respecto al punto de medición. Conocidas las distancias, se determina fácilmente la propia posición relativa respecto a los satélites. También se consigue una exactitud extrema en el reloj del GPS, similar a la de los relojes atómicos que llevan a bordo cada uno de los satélites.

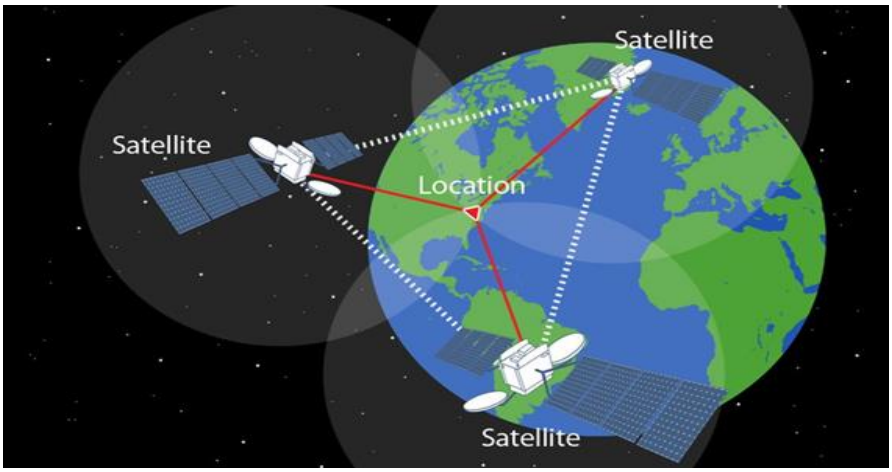
2.4.2.2 Funcionalidad de GPS en Android

El GPS es una de las piezas más importantes en los dispositivos Android ya que es un sistema capaz de localizar en cualquier parte del mundo, y que resulta muy útil en combinación a aplicaciones de mapas o de indicaciones. Tiene sus limitaciones, pero es perfecto para las necesidades de muchos.

GPS depende en que cada satélite en la constelación transmita su posición exacta y una señal de tiempo extremadamente precisa a los recibidores en la tierra. Dada esta información, los receptores GPS pueden calcular su distancia al satélite, y combinando esta información de cuatro satélites, el recibidor puede calcular su posición exacta usando un proceso llamado trilateración.

A partir de esa base, cuando el smartphone quiera localizarse, se conectará a esta red y conectará con la mayor cantidad de satélites posibles, obteniendo una serie de datos, y utiliza la triangulación inversa -averiguar la distancia de cada satélite respecto a la posición del dispositivo para para ser situado en el mapa.

Ilustración 6. Funcionalidad de GPS en Android (Tomado de [7])



No es el único sistema que existe para localizar en la Tierra, dado que el GPS es un desarrollo del Departamento de Defensa de los EEUU: GLONASS es el actual nombre del comenzado por la Unión Soviética y continuado por Rusia, Galileo es la alternativa europea y el único destinado al uso civil, y Beidou-barra-Compass es el proyecto de la República Popular de China que da servicio a China y países vecinos.

2.4.3. Web Service

Es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos.

El término Web Services describe una forma estandarizada de integrar aplicaciones WEB mediante el uso de XML, SOAP, WSDL y UDDI sobre los protocolos de la Internet. XML es usado para describir los datos, SOAP se ocupa para la transferencia de los datos, WSDL se emplea para describir los servicios disponibles y UDDI se ocupa para conocer cuales son los servicios disponibles. Uno de los usos principales es permitir la comunicación entre las empresas y entre las empresas y sus clientes. Los Web Services permiten a las organizaciones intercambiar datos sin necesidad de conocer los detalles de sus respectivos Sistemas de Información.

A diferencia de los modelos Cliente/Servidor, tales como un servidor de páginas Web, los Web Services no proveen al usuario una interfaz gráfica (GUI). En vez de ello, los Web Services comparten la lógica del negocio, los datosy los procesos, por medio de una interfaz de programas a través de la red. Es decir, conectan programas, por tanto son programas que no interactúan directamente con los usuarios.

Los desarrolladores pueden por consiguiente agregar a los Web Services la interfaz para usuarios, por ejemplo, mediante una pagina Web o un programa ejecutable, tal de entregarles a los usuarios una funcionalidad específica que provee un determinado Web Service.

2.4.2.1 Tecnología Web Services

Los Web Services están construidos con varias tecnologías que trabajan conjuntamente con los estándares que están emergiendo para asegurar la seguridad y operatibilidad, de modo de hacer realidad que el uso combinado de varios Web Services, independiente de la o las empresas que los proveen, este garantizado. A continuación, se detallan las tecnológias de web service:

2.4.3.1.1 XML

El XML (*E***x***tensible* **M***arkup* **L***anguage*), permite a los desarrolladores crear sus propios tags, habilitar definiciones, transmiciones, validaciones, e interpretación de los datos entre aplicaciones y entre organizaciones.

2.4.3.1.2 UDDI

El UDDI (*Universal* **D***escription,* **D***iscovery and* **I***ntegration)*, es un directorio distribuido que opera en la Web y permite a las empresas publicar sus Web Services, para que otras empresas los conozcan y los utilicen, opera de manera simultánea a las páginas amarillas.

2.4.3.1.3 SOAP

Es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. Básicamente SOAP es un paradigma de mensajería de una dirección sin estado, que puede ser utilizado para formar protocolos más complejos y completos según las necesidades de las aplicaciones que lo implementan.

Puede formar y construir la capa base de una "pila de protocolos de web service", ofreciendo un framework de mensajería básica en el cual los web services se pueden construir. Este protocolo está basado en XML y se conforma de tres partes:

- Sobre (envelope): el cual define qué hay en el mensaje y cómo procesarlo
- Conjunto de reglas de codificación para expresar instancias de tipos de datos
- La Convención para representar llamadas a procedimientos y respuestas.

El protocolo SOAP tiene tres características principales:

- Extensibilidad (seguridad y WS-routing son extensiones aplicadas en el desarrollo).
- Neutralidad (SOAP puede ser utilizado sobre cualquier protocolo de transporte como HTTP, SMTP, TCP o JMS).
- Independencia (SOAP permite cualquier modelo de programación).

“La arquitectura SOAP está formada por varias capas de especificación: MEP (Message Exchange Patterns) para el formato del mensaje, enlaces subyacentes del protocolo de transporte, el modelo de procesamiento de mensajes, y la capa de extensibilidad del protocolo. SOAP es el sucesor de XML-RPC, a pesar de que toma el transporte y la neutralidad de la interacción, así como el envelope / header / body, de otros modelos (probablemente de WDDX)”. [8]

Estructuera de SOAP: cuenta con una estructura definida en la especificación del protocolo, conformada por las siguientes partes:

- **Envelope (obligatoria)**: raíz que, de la estructura, es la parte que identifica al mensaje SOAP como tal.
- **Header:** es una herramienta para que los mensajes puedan ser enviados de la forma más conveniente para las aplicaciones. El elemento "Header" se compone a su vez de "Header Blocks" que delimitan las unidades de información necesarias para el header.
- **Body (obligatoria):** contiene la información relativa a la llamada y la respuesta.
- **Fault:** bloque que contiene información relativa a errores que se hayan producido durante el procesado del mensaje y el envío desde el "SOAP Sender" hasta el "Ultimate SOAP Receiver"

El modelo de procesado de SOAP: está definido como un sistema distribuido, en el que intervienen diferentes nodos. En un escenario básico, los nodos SOAP se comunican uno asumiendo el rol de SOAP Sender y SOAP Receiver. Aun así, la especificación define diferentes tipos de nodos en función del rol que asumen en el envío del mensaje:

- **SOAP Sender**: nodo que transmite un mensaje SOAP.
- **SOAP Receiver:** nodo que acepta un mensaje.
- **SOAP message path:** conjunto de nodos por los cuales debe pasar un mensaje SOAP, incluyendo el nodo inicial, cero o más nodos intermediarios y el SOAP Receiver definitivo.

- **Initial SOAP sender:** el "sender" que origina el mensaje y que es el punto de inicio del camino que seguirá el mensaje.
- **SOAP intermediary:** el intermediario actúa como SOAP receiver y como SOAP sender, ya que primero recibe el mensaje para después reenviarlo al siguiente nodo en el camino.
- **Ultimate SOAP receiver:** destino final del mensaje SOAP, es el responsable de procesarlo. Cabe destacar que el mensaje podría no llegar al receptor definitivo debido a que problemas en los intermediarios hagan que se pierda.

Un nodo SOAP puede actuar con uno o varios roles, cada uno de los cuales se encuentra definido mediante una URI conocida como el nombre de rol; los roles asumidos por un nodo son invariantes durante el envío de un mensaje, teniendo en cuenta la especificación el procesado individual de mensajes. Tal y como se ha comentado, una aplicación puede crear protocolos de comunicación más complejos como capas superiores sobre SOAP, pudiendo definir sus propios roles para poder cumplir con sus necesidades.

Casos de uso SOAP : La forma más habitual de utilizar el protocolo SOAP es mediante el patrón petición-respuesta con remitente SOAP y destinatario final SOAP, el cual es utilizado cuando los mensajes SOAP están predefinidos y únicamente se desea enviar una petición y consultar su valor de retorno.

No obstante, muchas veces este patrón no es suficiente, y es necesario establecer un intercambio múltiple de mensajes entre los nodos. La W3C define dos tipos de intercambios de mensajes SOAP para formar una conversación:

- **Intercambio de mensajes Conversacionales:** Permite redefinir la información de la petición. Estos intercambios pueden acabar comportándose como un patrón de mensajes de ida y vuelta.
- **Llamadas a Procedimientos Remotos:** Permite encapsular la funcionalidad de procedimientos remotos utilizando las ventajas de XML de extensibilidad y funcionalidad, por este motivo se ha definido en la especificación una

representación uniforme para realizar invocaciones y respuestas RPC mediante mensajes SOAP.

En ocasiones, es necesario el uso de intermediarios en las comunicaciones SOAP, la especificación SOAP 1.2 define dos tipos:

- **Intermediario redirector:** Se trata de un nodo SOAP, el cual redirige el mensaje SOAP a otro nodo SOAP según lo establecido en un bloque de encabezado que ha recibido el nodo destino o según el patrón de mensajes en uso.
- **Intermediario activo:** Realiza un procesamiento adicional del mensaje SOAP antes de redirigirlo, sin utilizar criterios descritos en el encabezado del mensaje o del patrón de mensajes en uso.

2.4.3.1.4 WSDL

Es un formato XML que se utiliza para describir servicios Web. La versión 1.0 fue la primera recomendación por parte del W3C y la versión 1.1 no alcanzó nunca tal estatus. La versión 2.0 se convirtió en la recomendación actual por parte de dicha entidad.

WSDL describe la interfaz pública a los servicios Web, está basado en XML y describe la forma de comunicación, es decir, los requisitos del protocolo y los formatos de los mensajes necesarios para interactuar con los servicios listados en su catálogo. Las operaciones y mensajes que soporta se describen en abstracto y se ligan después al protocolo concreto de red y al formato del mensaje.

Así, WSDL se usa a menudo en combinación con SOAP y XML Schema. Un programa cliente que se conecta a un servicio web puede leer el WSDL para determinar qué funciones están disponibles en el servidor. Los tipos de datos especiales se incluyen en el archivo WSDL en forma de XML Schema. El cliente puede usar SOAP para hacer la llamada a una de las funciones listadas en el WSDL además permite tener una descripción de un servicio web. Especifica la interfaz abstracta a través de la cual un cliente puede acceder al servicio y los detalles de cómo se debe utilizar.

Estructura del WSDL, compuesta por los siguientes elementos:

- **Tipos de Datos**<types>: Esta sección define los tipos de datos usados en los mensajes. Se utilizan los tipos definidos en la especificación de esquemas XML.
- **Mensajes**<message>: Aquí definimos los elementos de mensaje. Cada mensaje puede consistir en una serie de partes lógicas. Las partes pueden ser de cualquiera de los tipos definidos en la sección anterior.
- **Tipos de Puerto**<portType>: Con este apartado definimos las operaciones permitidas y los mensajes intercambiados en el Servicio.
- **Bindings**<binding>: se especifican los protocolos de comunicación usados.
- **Servicios**<service>: Conjunto de puertos y dirección de los mismos. Esta parte final hace referencia a lo aportado por las secciones anteriores.

PHP NuSOAP

Para el intercambio de información entre los servidores y la aplicación se utiliza PHP nuSOAP, el cual es una re escritura de SOAPx4, provisto por NuSphere, el cual se basa en estándares SOAP 1.1, WSDL 1.1 y HTTP 1.0/1.1. Este ha sido seleccionado por la difusión del mismo y la facilidad de implementación al ser una clase PHP.

CAPITULO III
ANALISIS, DISEÑO E IMPLEMENTACION DE LA APLICACIÓN MOVIL

3.1 ANÁLISIS

Dentro del análisis de la aplicación se consideró el análisis de requerimientos para la creación de la app, el desarrollo tanto a nivel funcional como no funcional de la aplicación y su integración con la plataforma con la que cuenta el Gobierno Provincial.

3.1.1 Requerimientos para la creación de la aplicación móvil Android

Una vez analizado el impacto de utilizar una aplicación móvil frente a la actividad manual en el proceso de levantamiento de datos, se procedió con el análisis de la aplicación en cuanto a su funcionalidad y su versatilidad. Para realizar esto se hizo un análisis de requerimientos de la aplicación móvil, y como esta podría interactuar con la información con la que consta actualmente el sistema de información turística, haciendo una relación entre los requerimientos que se tenía de la aplicación con las soluciones que se pueden dar, para este sistema, como se muestra a continuación en la Ilustración 7.

Ilustración 7. Análisis de requerimientos de la Aplicación Android

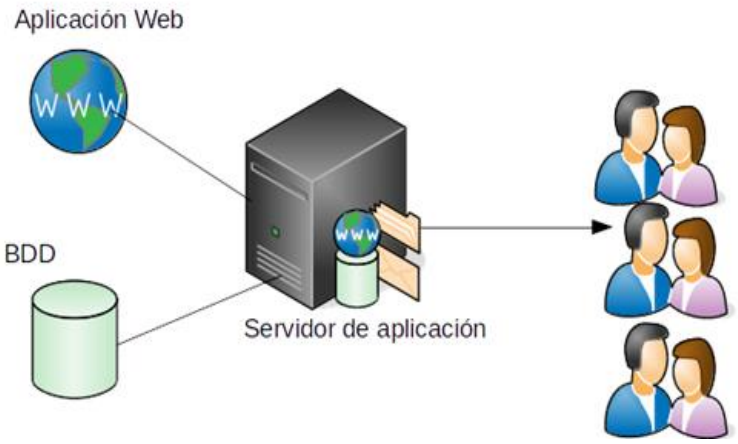
Elaborado por: El Autor

Como se puede apreciar en el gráfico del lado izquierdo los requerimientos sobre la aplicación y del lado derecho la manera en que estos se tenían previsto solventar, estos requerimientos, son de nivel de requerimientos de usuario final para evitar la carga técnica a los requirentes.

Realizado el análisis de requerimientos, de los requirentes se procedió a realizar un análisis de cuál sería la solución que mejor se ajuste a los requerimientos, por lo cual se empezó a realizar un análisis de la arquitectura de la aplicación.

En este caso se analizó la tecnología a utilizar y la tecnología disponible dentro del sistema de información turística que mantiene la Prefectura del Carchi, buscando la interoperabilidad entre los dos sistemas y de ser el caso, realizar las interfases para poder interoperar los sistemas, para lo cual se revisará la estructura de la aplicación turística, como se observa en la Ilustración 8.

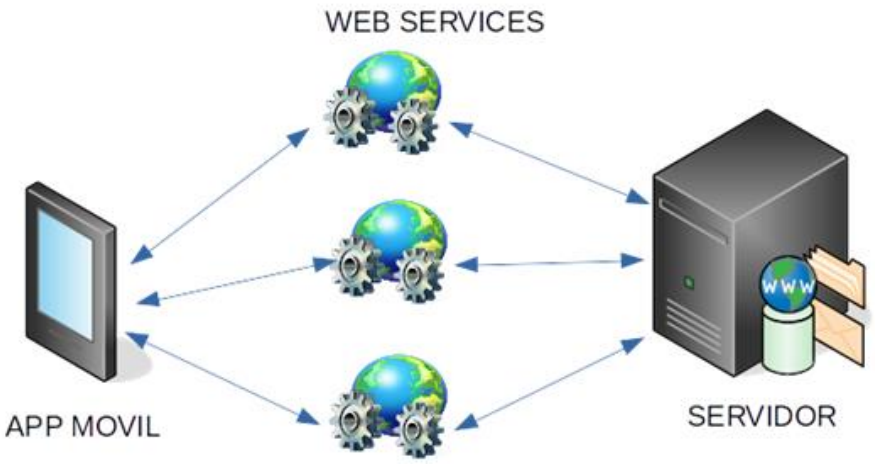
Ilustración 8. Diagrama del sistema de Turismo



Elaborado por: El Autor

Como se observa la aplicación de turismo es una aplicación web en la que se gestionan la aplicación y la base de datos a través de un servidor el cual sirve de interfaz para el acceso de la ciudadanía. Una vez analizada la estructura de la aplicación se procede a definir la estructura de la aplicación móvil, tal como se presenta en la Ilustración 9.

Ilustración 9. APP de levantamiento de información turística



Elaborado por: El Autor

La aplicación para poder tener un control sobre la información debe ser generada con una redundancia de la información en una base de datos, eso permite a futuro gestionar la información con el fin de poder realizar mejora en los servicios y tomar decisiones basados en datos concretos.

Una opción viable desde el análisis de requerimientos es el uso de servicios web que permitieran evitar la carga de información en los dispositivos móviles y poder gestionarlos en un servidor con las ventajas que esto conlleva para la gestión de información en una base de datos.

3.1.1.1 Requerimiento tecnicos de la app

- Se desarrolló una aplicación funcional utilizando los programas: IDE Android Studioy Sitio web “www.carchi.gob.ec./turistico”
- El lenguaje de programación que se utilizó para desarrollar la aplicación móvil utilizando IDE Android Studio fue Java, adecuado para Android.
- Como requisitos mínimos se requierió de un computador de escritorio o laptop con procesador CORE 2 de 2GHz, memoria de 4GB y 100GB de espacio en el disco duro, conexión a internet LAN.

- Sistema Operativo para servidor Centos versión 6.5
- Receptor GPS.
- Smartphone con GPS y de conexión a Internet a través de datos móviles o wiFi.

3.1.1.2 Requerimientos funicionales de la app

Los requerimientos funcionales del proyecto son los siguientes:

- Aplicación de tipo móvil
- Sistema operativo Android
- Autenticación contra servidor
- Interfaz de acceso gráfica
- Interoperabilidad con servidor implementado a través de web services
- Módulo de chat
  - ✓ Permitir chatear entre usuarios de la aplicación
  - ✓ Mensajeria en tiempo real
- Módulo de posicionamiento
  - ✓ Mostrar un mapa con la posición actual
  - ✓ Mostrar la posición de los usuarios de la aplicación
- Modulo de carga de información
  - ✓ Mostrar las fichas de los tipos de información que tiene la aplicación
  - ✓ Cargar la información alfanumérica y geográfica
  - ✓ Interactuar por servicios web

3.1.1.3 Requerimientos no funcionales de la app

Usabilidad: El uso de la aplicación móvil es sencillo ya que va a tiene una interfaz gráfica intuitiva, para que el usuario sepa lo que debe hacer, por lo cual se debe considerar lo siguiente:

- La aplicación va ha estar disponible en idioma español y utiliza los colores institucionales.

- El contenido de la app no debe ser alterado por los diferentes colores o imágenes utilizadas para su desarrollo.
- El tipo de letra debe ser legible de tamaño adecuado, esto puede variar según el tamaño de la pantalla del dispositivo móvil.
- La orientación del teléfono siempre será vertical en todas las ventanas.

Rendimiento: La aplicación debe ser capaz de ejecutarse y funcionar en un dispositivo móvil con un sistema operativo Android mínimo API16 (Android 4.1 Jelly Bean).

La aplicación debe responder en tiempo real para los usuarios que vayan a hacer uso de la misma

La cantidad de información que va a manejar la app será mínima, y está relacionada al envío y recepción de mensajes de texto mismos que se guardarán en una carpeta en la memoria interna del dispositivo móvil y al envío de datos desde la app hacia el servidor web.

Disponibilidad: La app estará disponible y ejecutará todas sus funcionalidades en un dispositivo móvil, siempre que se den las siguientes condiciones:

- Pantalla de 4” en adelante
- Sistema Operativo Android 4.1 o superior
- Disponer de servicio GPS con soporte A-GPS
- Habilitar la conexión a Internet (Wi-Fi o móvil)
- Contar con carga en la batería
- La aplicación debe estar disponible acorde a los SLA que maneja el Gobierno Provincial.
- La aplicación una vez instalada en el móvil no debe generar conflictos con otras apps instaladas en éste.
- La app cuando este en funcionamiento no va a colapsar en ningún momento de su utilización.

Seguridad: Para garantizar el acceso a la información se debe interactuar con el servidor a través de servicios web

3.2 DISEÑO

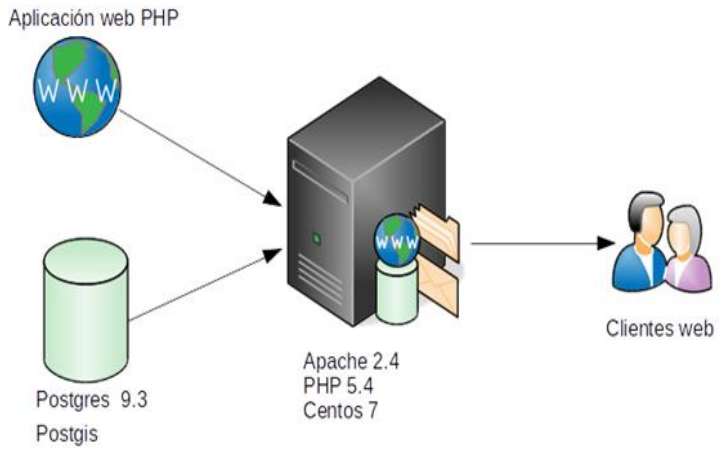
Dentro del diseño se analizó la arquitectura de la aplicación y los servicios de la aplicación para terminar en el modelamiento de los diagramas de clases.

3.2.1 Diagrama Arquitectónico

Una vez realizados los análisis de requerimientos y revisada la estructura de las aplicaciones es conveniente revisar qué tecnología se va a usar en la aplicación y como esta se va a relacionar con la aplicación de turismo.

Para conocer cómo se va a relacionar con la aplicación de turismo, es necesario conocer la arquitectura de la aplicación de turismo, la cual se detalla en la Ilustración 10.

Ilustración 10. Arquitectura de web Turismo



Elaborado por: El Autor

Como se aprecia la aplicación de turismo es una aplicación web que se puede acceder a través de un servidor que está escuchando en el puerto 80 con las siguientes características:

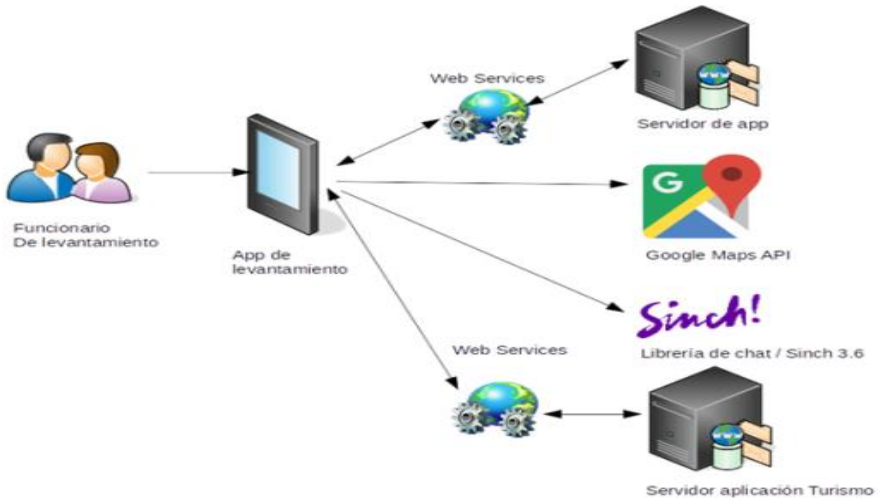
- **Sistema Operativo:** El sistema operativo que monta el servidor es un sistema operativo Linux, una distribución llamada CentOS, en la versión 7.
- **Servidor de web:** El servidor web utilizado para la aplicación de turismo es el un servidor web Apache en la versión 2.4.
- **Lenguaje de programación:** El lenguaje de programación dentro de la aplicación web de turismo es PHP en su versión 5.4.
- **Base de datos:** La base de datos utilizada en esta aplicación es una base de datos relacional llamada Postgresql en su versión 9.3 con un módulo espacial llamado postgis.

Analizada la estructura de la aplicación de turismo se procede a realizar la arquitectura de la aplicación móvil, tomando en cuenta los siguientes factores:

- Necesidades de la aplicación
- Diagramas de sistemas web
- Arquitecturas de la aplicación web

Con estos factores se diseñó la arquitectura de la aplicación que se presenta a continuación en la Ilustración 11.

Ilustración 11. Arquitectura de aplicación móvil



Elaborado por: El Autor

La aplicación móvil objeto de este caso de estudio tiene un conjunto de tecnologías por detrás que se detallan en el diagrama anterior donde se puede destacar que su funcionalidad tiene cuatro pilares a mencionar: autenticación y almacenamiento de sesiones, chat, geoposicionamiento y levantamiento de catastro.

Previo al desglose de cada uno de estos componentes, se señala las diferentes tecnologías utilizadas en el desarrollo de la app.

Sistema operativo. - La aplicación se montó sobre smartphones con sistema operativo Android, dado las ventajas que tiene este siendo software libre con ventajas como la interoperabilidad, costo de equipos, licencias de uso y distribución, facilidad de instalar las aplicaciones en los dispositivos móviles, entre otros.

App: La aplicación generada va a ser producto de un desarrollo basado en los requerimientos puntuales que se tienen y los diferentes diagramas provistos.

Web Service: La aplicación realiza las interacciones con los distintos servidores a través de servicios web, los cuales están provistos por los servidores de la aplicación y de la plataforma de turismo y consumidos por la aplicación.

Google Maps API: Para el desarrollo y el despliegue de información de la aplicación en mapas se utilizó el API de Google Maps, el cual permite acceder de una forma centralizada a través del Google API Client.

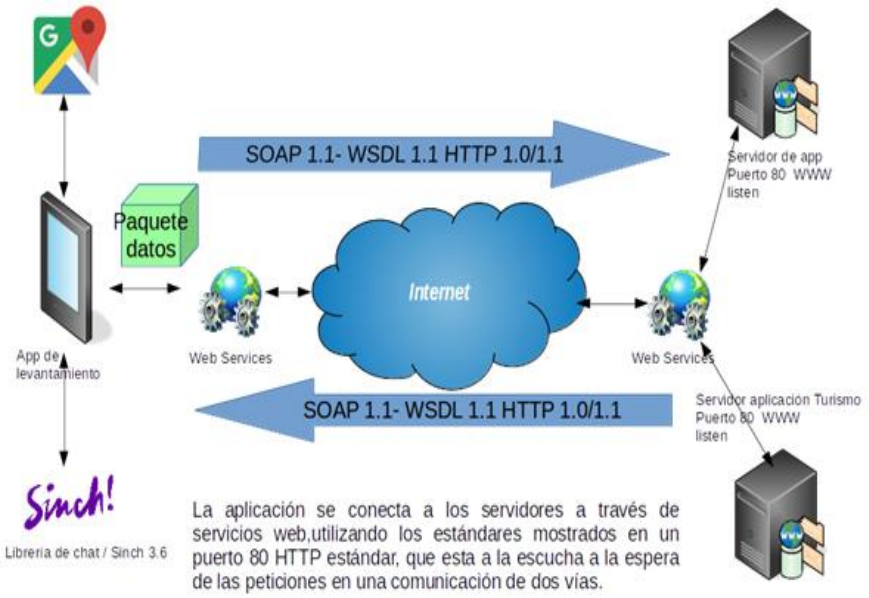
Librería de Chat: Para poder tener la funcionalidad de chat se trabajaó con un API, llamado Sinch el cual permite acceder a chat a través de la configuración de la misma dentro de la aplicación.

Sinch ofrece una plataforma para la comunicación en tiempo real a través de Internet. Se compone de diferentes kits de desarrollo de software como son los SDK Sinch que se integra con el teléfono inteligente o una aplicación web basada en la nube y los servicios de back-end. En conjunto, permiten la comunicación de voz basado en la aplicación.

3.2.2 Diagrama de Red de la Aplicación

Para entender de una mejor manera como va a interoperar la aplicación se pone a consideración la Ilustración 12.

Ilustración 12. Diagrama Red App Móvil



Elaborado por: El Autor

Como se muestra en el diagrama la aplicación se va a conectar con los servidores web por medio de internet. Para garantizar la conexión entre los dos y la disponibilidad de servicio se va a ocupar servicios web que trabajen en los servidores web.

La aplicación hace uso de los servicios web en dos momentos cruciales para el funcionamiento de la misma, el primero sucede al abrir la aplicación, está pide una autenticación la cual se va a validar en el servicio web provisto para la autenticación por el servidor de la aplicación.

La comunicación se realiza desde un puerto aleatorio del lado de la app y se conecta al puerto 80 de la aplicación, mediante los servicios web, el envió se realiza mediante un HTTP Request del lado de la aplicación la que para realizar esto hace un parseo de los datos provistos en la aplicación y los envía. El servidor responde de la misma manera con HTTP

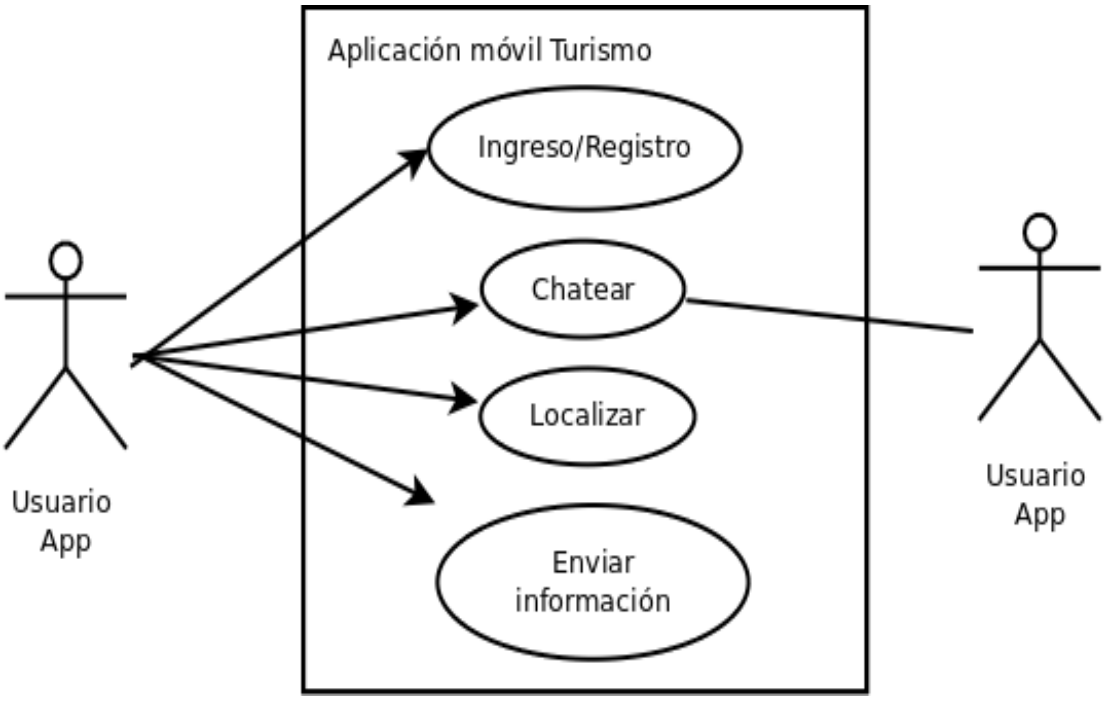
Response donde devuelve la información solicitada. Previo a la devolución de la información requerida el servicio web se encarga de realizar un procesamiento, en este caso es una validación contra una base de datos que maneja las autenticaciones.

El segundo momento en que se utiliza es cuando se envía la información levantada a través de la app móvil, está parsea la información y la envía hacia el servidor a través de la función de carga de información. Aquí el servidor recoge esta información y la procesa acorde al requerimiento y la manda a persistir en una base de datos la cual permite que se tenga información actualizada de una manera ágil.

3.2.3 Diagrama de caso de uso general de la aplicación móvil

Para el presente desarrollo se tiene un diagrama de caso de uso donde se muestra las operaciones disponibles para los usuarios dentro de la aplicación y como estos interactuan con otros usuarios de la aplicación.

Ilustración 13. Aplicación móvil de turismo



Elaborado por: El Autor

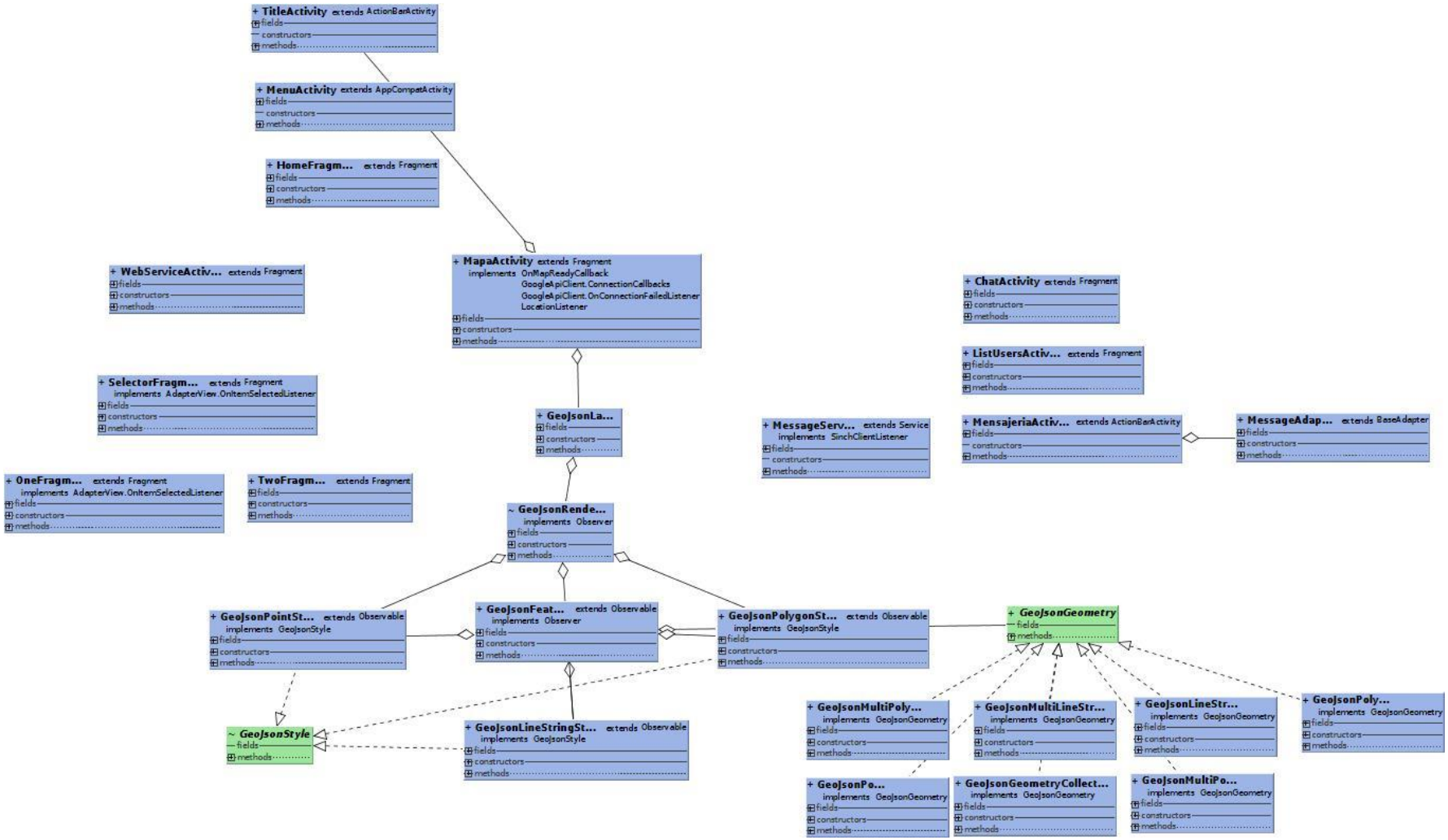
El usuario estándar de la aplicación tiene cuatro opciones dentro de la aplicación a describir:

- **Ingreso/registros:** Primera acción a realizar por los usuarios, teniendo la opción de registrarse como un usuario nuevo o ingresar si ya es un usuario de la aplicación.
- **Chatear:** Los usuarios ingresados pueden hacer uso del chat para poder interactuar con otros usuarios de la aplicación.
- **Localizar:** Por medio de esta sección el usuario puede conocer su posición en el mapa, así como la posición de los usuarios que se encuentran registrados en la aplicación.
- **Enviar información:** Esta sección permite acceder a los formularios de los tipos de información que la aplicación puede cargar al servidor.

3.2.4 Diagramas de clases

Este diagrama permite identificar las clases dentro de la aplicación, así como sus distintas relaciones en el caso de que las tengan tal como se indica en la ilustración 14.

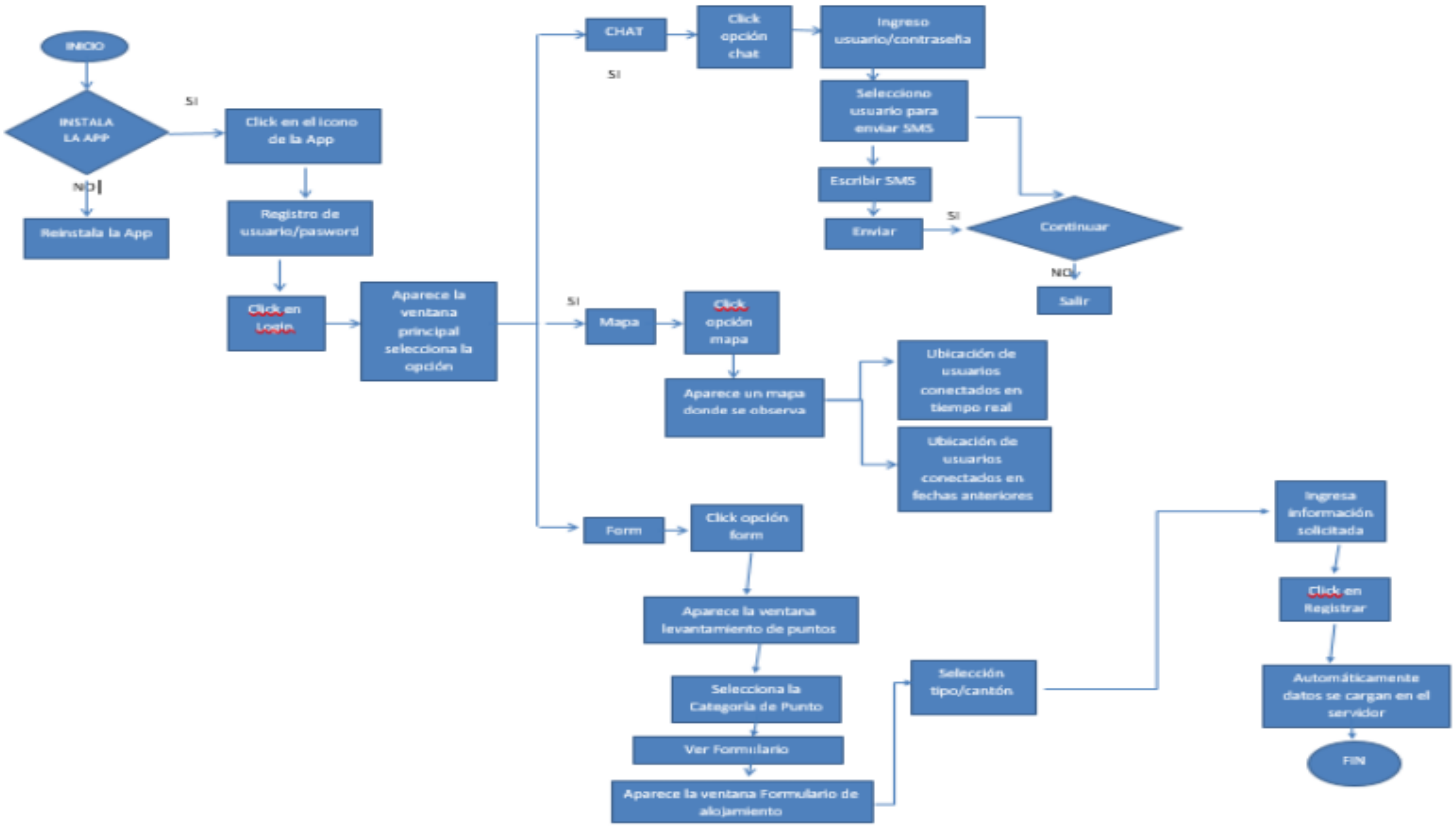
Ilustración 14. Diagrama de Clases



Elaborado por: El Autor

3.2.5 Diagrama de Flujo de la aplicación móvil

Ilustración 15. Diagrama de Flujo



Elaborado por: El Autor

3.3 IMPLEMENTACIÓN

3.3.1 Desarrollo de la aplicación móvil

3.3.1.1 Wireframe de la aplicación

Para tener un mejor manejo sobre el desarrollo de la aplicación se realizó un wireframe de la misma para evidencia la estructura que va a tener la aplicación desarrollada como se muestra en las siguientes imágenes:

Login. - Será la pantalla que da la bienvenida, este contiene el logeo del usuario en la aplicación

Ilustración 16. LOGIN

Menú. - Una vez que se ha accedido a la aplicación se tiene acceso al menú de la misma que cuenta con tres opciones, el chat para interactuar con otros usuarios, el mapa que despliega mi posición y la de los otros usuarios y form, la sección donde se va a ingresar la información a levantar.

Ilustración 17. MENU

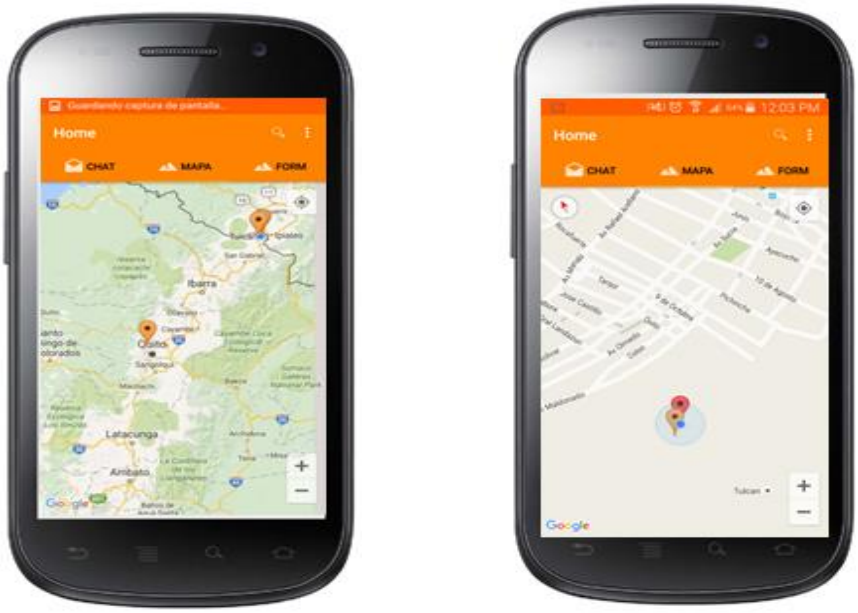
Chat. - Esta sección consta de tres secciones la primera es la que se visualiza al iniciar la aplicación, la segunda es la ventana de selección de los usuarios con quien se va a interactuar y la tercera la del chat propiamente dicho, como se muestra a continuación.

Ilustración 18. CHAT ½

Ilustración 19. CHAT 2/2

Mapa. - En esta sección se despliega la ubicación de los usuarios de la aplicación asi como la ubicación del dispositivo, desplegando la información del usuario y la fecha de conexión.

Ilustración 20. MAPA



Formulario. - En esta sección se realiza el levantamiento de información turística de la provincia consta de dos pantallas, las cuales permiten seleccionar el tipo de información que se va a levantar y enviarla al servidor para su procesamiento.

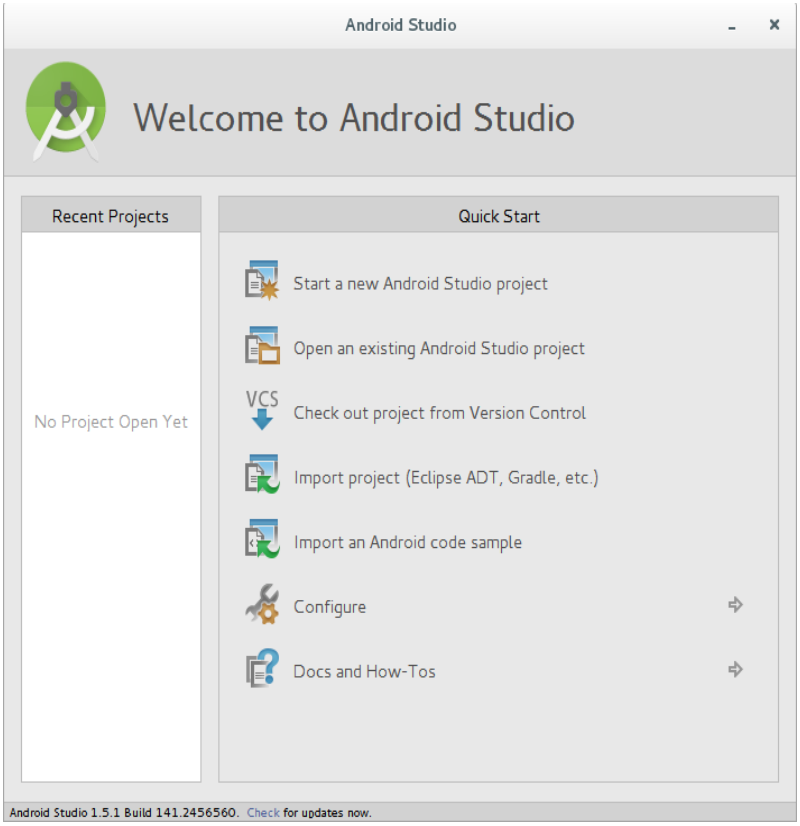
Ilustración 21. FORM ½

Ilustración 22. FORM 2/2

3.3.1.2 Creación de la aplicación móvil en Android Studio

Para iniciar con la programación de de la aplicación lo primero a realizar es generar el proyecto en el IDE seleccionado (Android Studio), usando como base los wireframes que se han elaborado previamente. Para arrancar con esto se debe abrir el Android Studio y aparecerá una imágen como la que se muestra a continuación en la Ilustración 23.

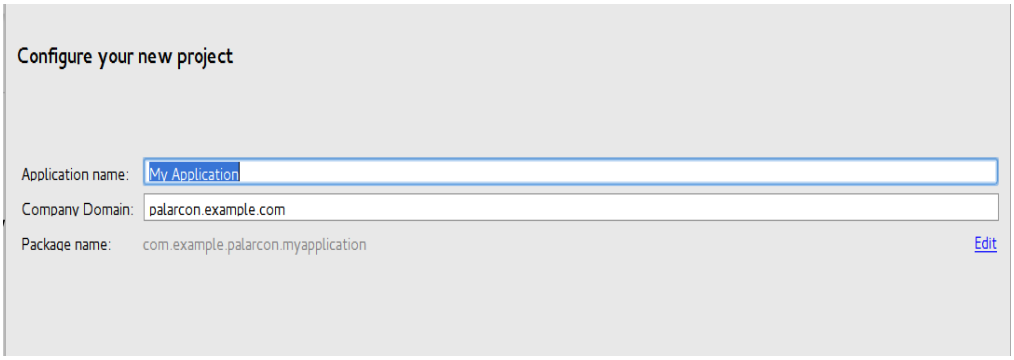
Ilustración 23. Iniciando un proyecto en Android Studio



Elaborado por: El Autor

Aquí se debe seleccionar **Start a new Android Studio project** para arrancar la creación del proyecto, donde va a pedir el nombre del proyecto en la sección **Application Name** y el dominio de la compañía en **Company Domain**, como se muestra en la Ilñustración 24.

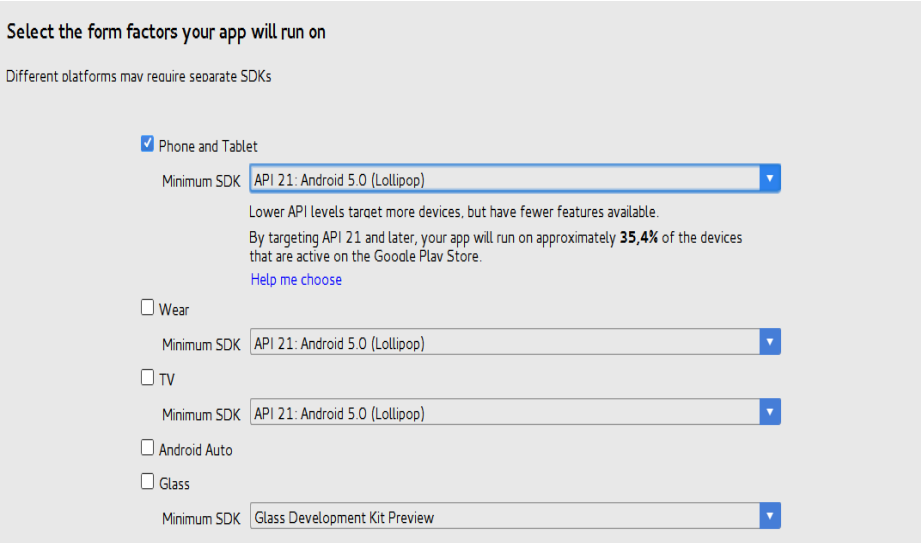
Ilustración 24. Configuración nuevo proyecto



Elaborado por: El Autor

A continuación, se va a preguntar para qué dispositivos se va a generar la aplicación y la versión de android que se va a trabajar, si se requiere de una gran compatibilidad se puede usar la API 15, que garantice más compatibilidad o una versión más actual que brinde más funcionalidades como material design, como se ve en las Ilustración 25.

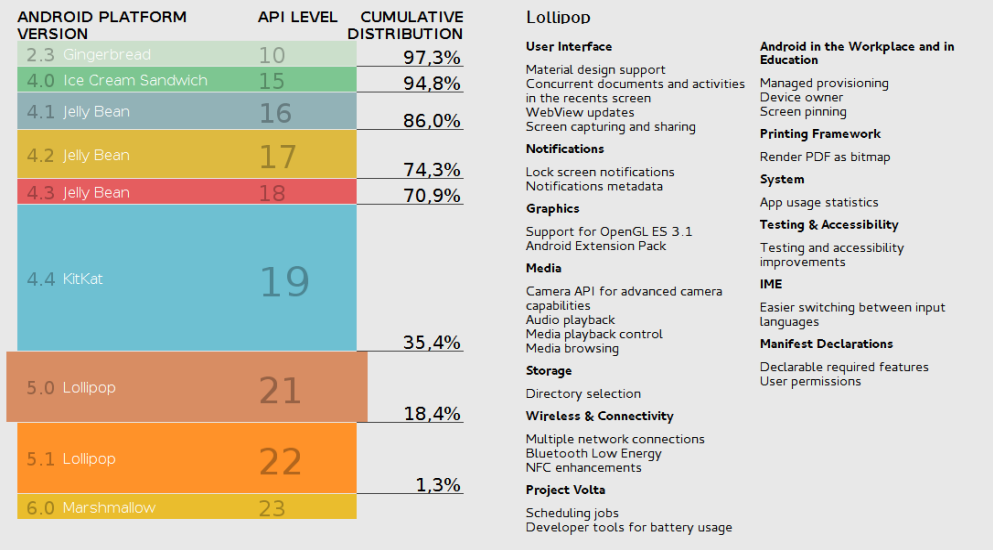
Ilustración 25. Selección dispositivo para generar la aplicación



Elaborado por: El Autor

Seleccionada la versión el IDE no permite seleccionar un tipo de layout de un asistente, que tiene una variada selección que puede ser elegido en función de la versión de android que se haya seleccionado como se observa en la Ilustración 26.

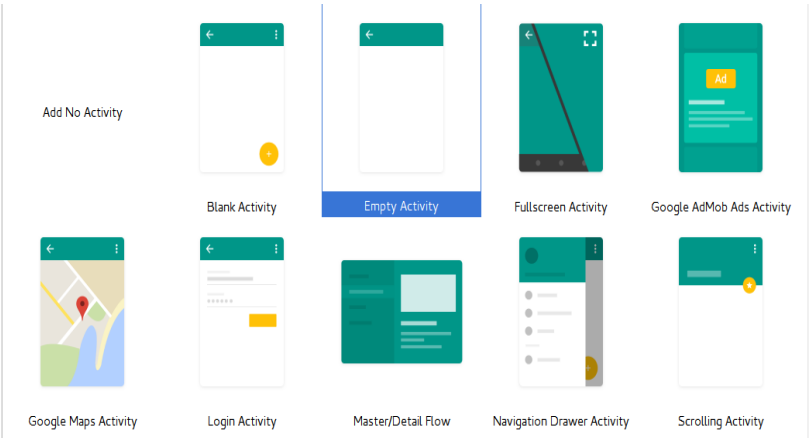
Ilustración 26. Selección de IDE



Elaborado por: El Autor

Una vez seleccionado el tipo de actividad se debe crear el nombre de la actividad que se va a generar, como se muestra en la ilustración 27.

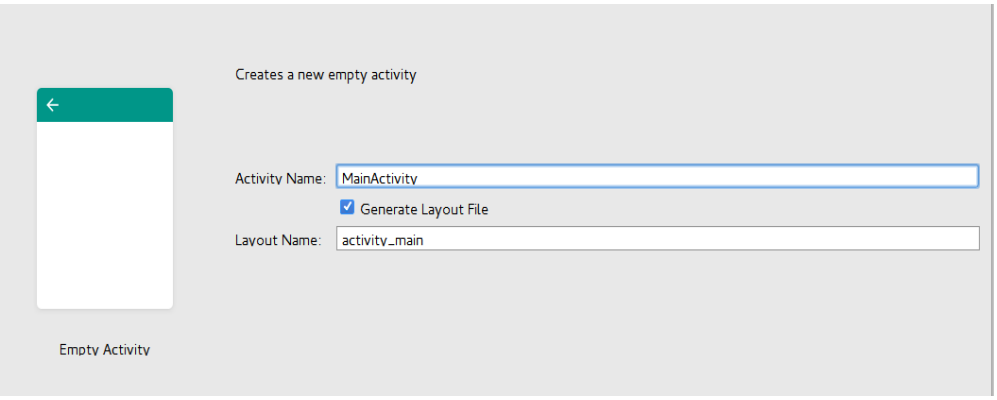
Ilustración 27. Creación del Nombre de actividad



Elaborado por: El Autor

Con esto se termina el proceso de generación del proyecto y se puede empezar a generar el código necesario para que la aplicación funcione.

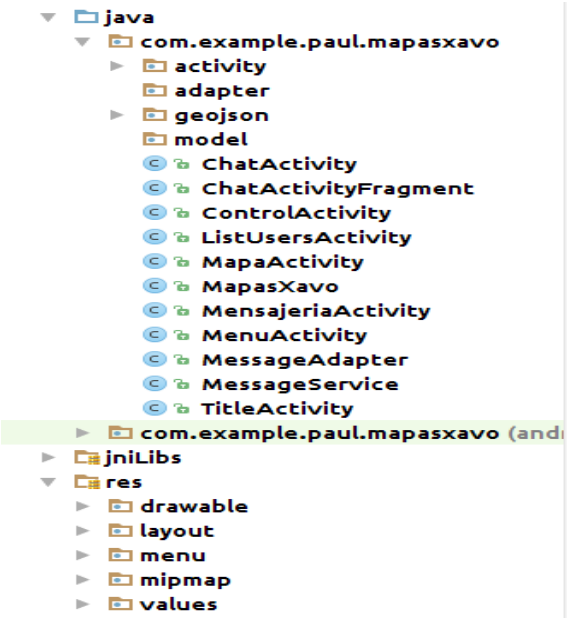
Ilustración 28. Generación de código



Elaborado por: El Autor

Finalizando la configuración del proyecto Android mismo que está compuesta por recursos y clases, los cuales están estructurados en Android Studio bajo carpetas como se muestra a continuación en la Ilustración 29.

Ilustración 29. Estructura de la aplicación Android



Elaborado por: El Autor

Las carpetas están definidas de la siguiente manera:

Java: Donde se almacenan las clases java de la aplicación.

Res: Donde se almacenan los recursos de la aplicación que están a su vez almacenados en las siguientes subcarpetas:

- **Drawable. -** Aquí se almacenan las imágenes que van a utilizar en la aplicación.
- **Layout. -** Aquí se almacenan los layouts en formato xml
- **Mipmap. -** Aquí se almacena el icono de la aplicación
- **Values. -** Aquí se almacena los valores de los strings de la aplicación tanto en el lenguaje nativo como para internacionalización.

3.3.1.3 Integrando Google Maps al desarrollo de la aplicación Android para el caso de estudio

Para iniciar con la integración del API de Google Maps dentro de la aplicación se requiere cumplir con ciertos requisitos dentro del ambiente de desarrollo como son los siguientes:

- IDE de desarrollo (Android Studio)

- Tener instaladas las últimas versiones del SDK Tools de Android
- Tener instaladas las librerías de soporte de Android
- Tener instalado Google Play Services API

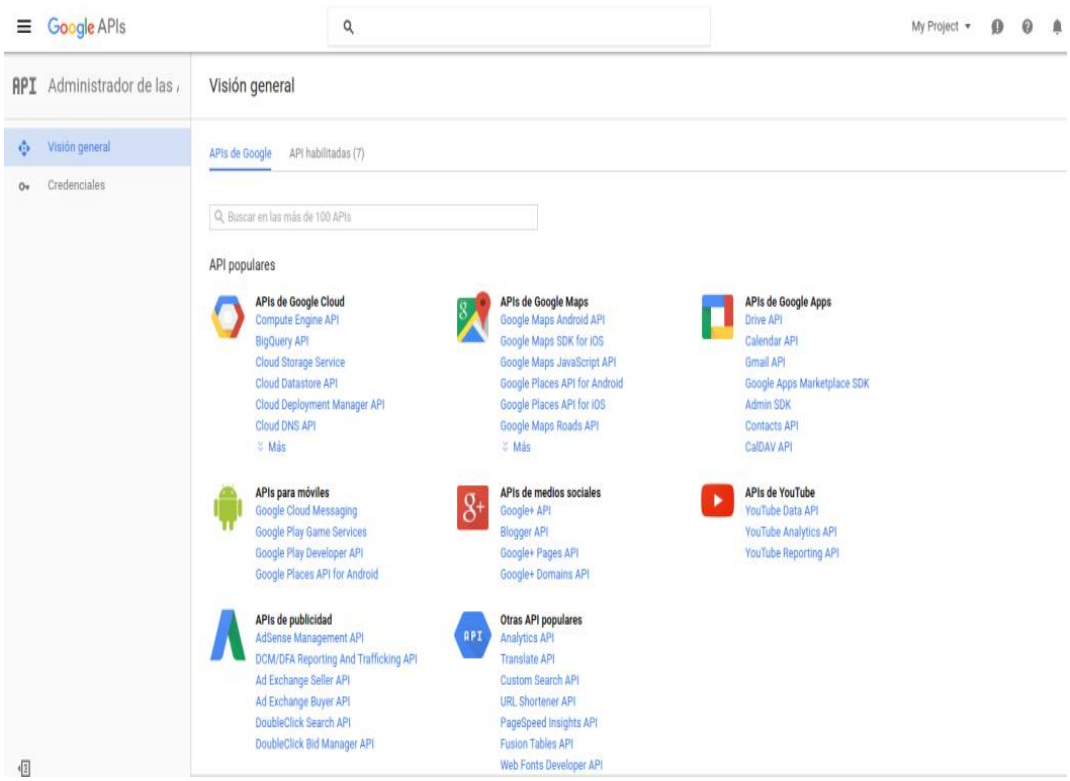
Cumplidos con estos requisitos se requiere agregar una nueva actividad de tipo Google Maps Activity al proyecto (clic derecho sobre el proyecto-> New -> Activity-> Gallery-> Google Maps Activity) se selecciona el nombre que se desea generar la nueva actividad respetando la convención de que las actividades deben contener la palabra Activity en el nombre y se da click en finish.

Creada la activity dentro de la aplicación se genera un archivo llamado **google_maps_api.xml**, este archivo contiene la clave para utilizar el Google Maps API, a continuación, se describe el proceso para generar una clave.

3.3.1.3.1 Generar una clave de Google Maps API

- Abrir el archivo google_maps_api.xml de la aplicación, para obtener las credenciales de la aplicación.
- Abrir la consola de administración de desarrolladores a través del enlace https://console.developers.google.com/ (se requiere estar logueado con el usuarios de gmail).

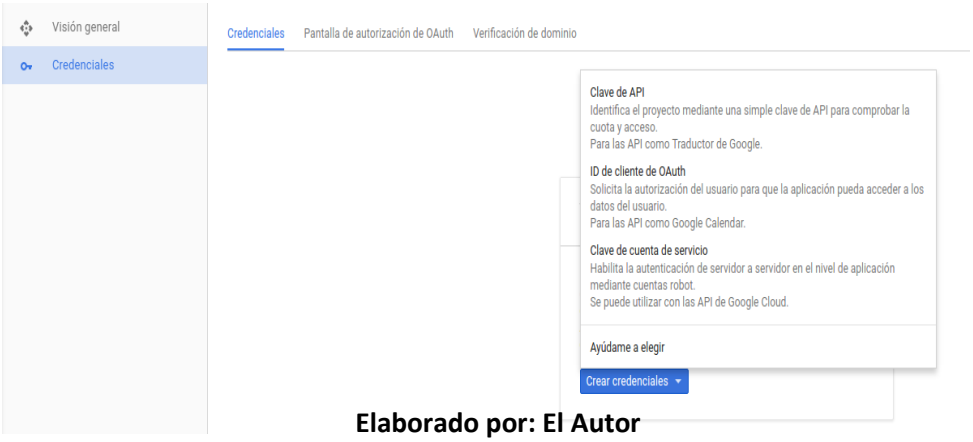
Ilustración 30. Generar una clave de Google Maps API



Elaborado por: El Autor

- Dentro de esta consola se genera un nuevo proyecto desde el combo que dice My Project en la parte superior derecha.
- Dentro del proyecto creado se selecciona las credenciales, y la clave del API en la opción crear credenciales

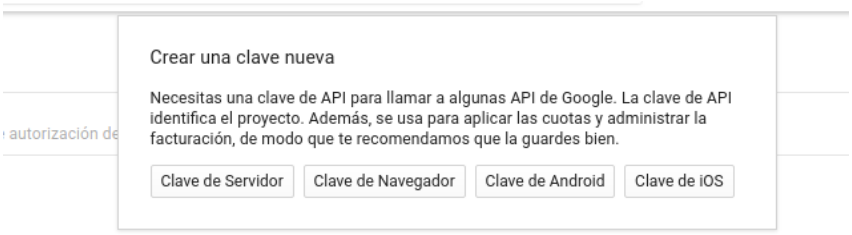
Ilustración 31. Selección de credenciales



Elaborado por: El Autor

- Dentro de esto seleccionar la clave de Android

Ilustración 32. Selección clave de Android



Elaborado por: El Autor

- Se selecciona la opción añadir nombre de paquete y huella digital
- Se pega aquí el valor que contiene el archivo generado en el archivo **google_maps_api.xml** (se encuentra como una huella SHA-1 en el archivo junto con el nombre del paquete) y se clickea en generar
- Copiar la clave de API en el archivo **google_maps_api.xml** reemplazando el texto YOUR KEY HERE por la clave que debe empezar por **Aiza.**

3.3.1.4 Generado el key de Google Maps

3.3.1.4.1 Configurando el mapa

La actividad que contiene el mapa implementa una interfaz llamada **OnMapReadyCallback la** cual extiende la funcionalidad de la clase Activity agregando la función **OnMapReady**, que es la encargada de lanzar la funcionalidad del mapa cuando se ejecute.

Dentro de esta función se puede cargar la información inicial para el mapa como los controles y la ubicación del mapa a desplegar, a continuación, se detalla las funciones utilizadas para realizar estos procedimientos dentro de la aplicación utilizando una función llamada **setupMap ()** de tipo void, tal como se observa en el ANEXO 1.

3.3.1.4.2 Obteniendo la ubicación del usuario

Android permite tener la ubicación del usuario dentro de la aplicación basado en algunos requerimientos y criterios que se deben cumplir previamente, a continuación, se detalla cómo se va a utilizar los servicios.

- Configurar Google Play Services, esto se realizó en los primeros pasos razón por la cual no se requiere habilitarlos de nuevo.
- Se deben especificar los permisos de la aplicación para acceder a la ubicación esto se logra mediante los permisos que se le otorgan a la misma a través del archivo **AndroidManifest.xml con** las siguientes sentencias tal como s indica en el ANEXO 2.

3.3.1.5 Integrando chat a la aplicación

La aplicación para el manejo del chat está usando una librería para mensajería instantánea llamada Sinch, que funciona como una API, para proveernos esta funcionalidad en la App. Esta aplicación es muy utilizada por aplicaciones como EasyTaxi, Tango, Uber entre las más conocidas.

Sinch utiliza un servicio llamado Parse para trabajar el cual es un BAAS (Backend as a Service).

Para integrar el chat a la aplicación se deben seguir los pasos que se señalan en el ANEXO 3.

3.3.1.6 Integrando servicios web a la aplicación

Para que la aplicación pueda interactuar con el servidor de turismo se ha optado por el consumo de servicios web, dado que es la solución más viable para realizar interacción entre las dos tecnologías usadas en las aplicaciones (PHP en la aplicación web y Java en la aplicación Android). Como el sistema web de Turismo no tenía inicialmente esa visión se vio

la necesidad de generar los servicios web en el servidor de turismo, usando la misma tecnología que se utilizó en el servidor de la aplicación para persistir y consultar los usuarios y las posiciones de los mismos.

Para describir la integración se la va a ver desde dos lados el primero la generación de los servicios web en el servidor y la segunda el consumo de estos en la aplicación cliente.

3.3.1.7 Generando los servicios web en el servidor

Los servicios web son creados a partir de una librería que se llama NuSoap la cual está disponible para PHP, a continuación, se describe el proceso de implementación y consumo de los mismo en el servidor, está se ha descargado en un carpeta llamada lib dentro del servidor de la aplicación a continuación se muestra su uso en la generación de los servicios publicados, ANEXO 4.

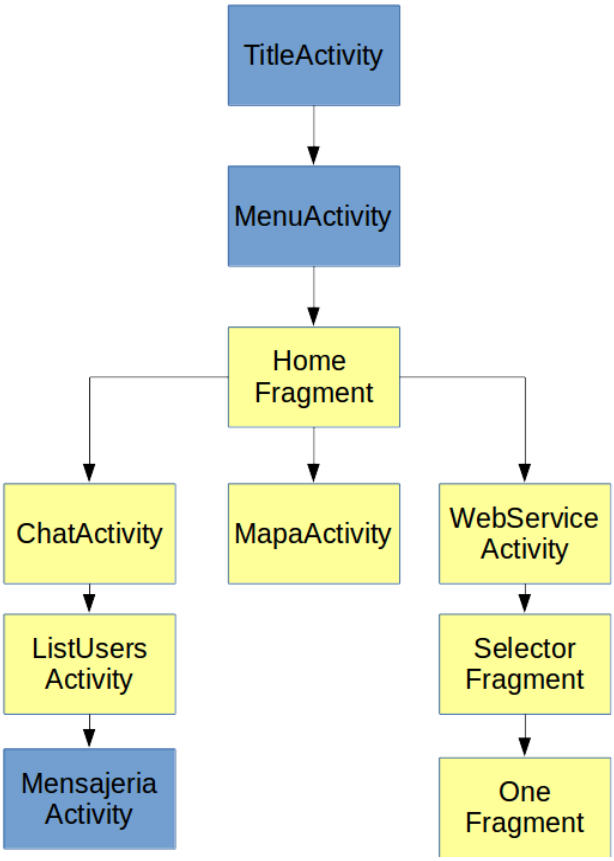
3.3.1.8 Consumiendo los servicios web del cliente

Para el consumo de los servicios generados en la aplicación se utiliza una librería llamada Ksoap2 la cual facilita el consumo de los servicios en los diferentes layouts que se quieran utilizar. En el ANEXO 5 se describe el proceso para el consumo desde la utilización de la libería.

3.3.1.9 Estructura gerárquica de la aplicación

Para un mejor entendimiento del contenido de la aplicación se pone a consideración el siguiente diagrama de la estructura jerárquica de la aplicación donde constan los componentes (activites y fragmentes) que se encuentran contenidos en la misma.

Ilustración 33. Aplicación de levantamiento de información turística



En este diagrama se muestran de color amarillo los framework y de color azul las activities en el orden de aparición de los mismos dentro de la aplicación

CAPITULO IV
INSTALACION Y EJECUCION PRUEBAS DE LA APLICACIÓN EN DISPOSITIVOS MOVILES

4.1 INSTALACIÓN

4.1.1 Requisitos para la instalación

La aplicación ha sido probada desde versiónes 4.1 Jelly Bean y posteriores mostrando un funcionamiento correcto, en los casos de prueba. Se requiere tener actualizados los servicios de Google Play para poder trabajar de una manera correcta.

Dado que la aplicación va a ser instalada fuera del Google Play se lo puede realizar de dos maneras:

- Corriendo la aplicación y subiendola desde Android Studio, lo cual requiere tener un perfil de desarrollador habilitado en el equipo o instalar.
- Descargar el ejecutable para lo cual se debe dar permiso a orígenes desconocidos dentro del equipo.

4.2 FUNCIONAMIENTO DE LA APLICACIÓN MÓVIL

Para que la aplicación móvil funcione en un Smartphone y poder utilizarla para el levantamiento de información y georreferenciación se debe seguir los pasos que se detallan acontuación:

Paso1. Solicitar la autorización al encargado del manejo de la app en la Prefectura del Carchi para utilizar la app.

Paso 2. Instalar la app en el dispositivo móvil, recordando que su versión no debe ser inferior a 4.1.

Paso 3. Una vez instalada la aplicación se muestra una imagen de android en la pantalla del Smartphone, se da click para poder ingresar.

Paso 4. Registrar el usuario y el pasword

Paso 5. Dar click en loginque **s**erá la pantalla que nos da la bienvenida.

Paso 6. Una vez que se ha accedido a la aplicación se tiene acceso al menú de la misma que cuenta con tres opciones: Chat, Mapa, Form, donde se selecciona la opción reuqerida.

Paso 7. Si dá click en la opción chat debe ingresar nuevamente el usuario y la contraseña.

Paso 8. Seleccionar los usuarios con quien se va a interactuar

Paso 9. Una vez seleccionado el usuario se escribe el mensaje y se da click en enviar.

Paso 10. Si su desición es ya no enviar mas mensajes da click en salir.

Paso 11. Si selecciona la opción mapa, usted puede observar que se despliega un mapa en el cual se puede observar la ubicación de los usuarios de la aplicación asi como la unicación del dispositivo que hace la petición, desplegando la información del usuario y la fecha de conexión.

Paso 12. Si selecciona la opción Form, aparece la ventana levantamiento de puntos, en la cual se selecciona la categoría de puntos.

Paso 13. dar click en ver formulario, donde aparece un formulario de acurdo a la categoria de puntos selecionada en el paso anterior.

Paso 14. Seleccionar tipo y cantón, y se ingresa la información solicitada que va a ser levantada in situ.

Paso 15. Dar click en registrar donde se enviará la información al servidor para su procesamiento.

4.3 EJECUCION DE PRUEBAS

4.3.1 Prueba de usabilidad

El objetivo de esta prueba es poder verificar si la manipulación de la aplicación es compleja o no, dicha prueba se la realizó tomando como muestra a un grupo multidiciplinario de 6 técnicos de diferentes edades de la Prefectura del Carchi que desarrollan su trabajo en campo. Para el desarrollo de la prueba se utilizó dos Smartphone: Samsung Galaxy S5 y Samsum Galaxy S4 conectado a una red Wi-Fi, además se plantearon tres preguntas sobre la ejecución de la aplicación, sobre las cuales se obtuvo el siguiente cuadro con los resultados que se indican a continuación:

Tabla 7. Manipulación de la aplicación

Númer de Técnico	Edad de los técncos encuestados	Complejidad en el manejo de la aplicacón Alta, Media, Baja	Ingreso de información en la aplicación Fácil, Difícil	Tiempo de carga de información a la base de datos Rápido, Medio, Lento	Presentación gráfica de la aplicación Excelente, Buena, Mala
1	32	Baja	Fácil	Rápido	Excelente
2	28	Baja	Fácil	Rápido	Excelente
3	39	Baja	Fácil	Rápido	Excelente
4	36	Baja	Fácil	Rápido	Excelente
5	24	Baja	Fácil	Rápido	Excelente
6	58	Baja	Fácil	Rápido	Excelente

Elaborado por: El Autor

Con los resultados obtenidos de los técnicos que apoyaron con el desarrollo de esta prueba, se concluye que el manejo de la aplicación no es complejo al igual que no lo es el ingreso de la información en ella solicitada, además manifestaron que el teimpo de carga de la información generada en la aplicación a la base de datos es rápido y que la presentación gráfica de la aplicación en general es excelente. Con lo anteriormente señalado se concluye que la aplicación cuenta con ventanas que son comprensibles, amigables e intuitivas.

4.3.2 Prueba de funcionalidad

Esta prueba se la realizó con el objetivo de corroborar el cumplimiento de los requerimientos establecidos previamente en el diseño de la aplicación móvil. Para lo cuál se trabajó con la aplicación en un Smartphone Galaxy S5:

Tabla 8. Funcionalidad de la aplicación

Requerimientos de la Aplicación	Cumplimiento
Disponer de servicio GPS	SI CUMPLE
Contar con conexión a Internet (Wi-Fi o móvil)	SI CUMPLE
Pantalla de 4” en adelante	SI CUMPLE
Sistema Operativo Android 4.1 o superior	SI CUMPLE
No genera conflictos con otras apps	SI CUMPLE
No colapsa en ningún momento de su utilización	SI CUMPLE
No cuenta con ventas deslizables	SI CUMPLE

Elaborado por: El Autor

Según la prueba realizada se concluye que los requerimientos básicos solicitados para el funcionamiento de la aplicación son necesarios para que la app funcione correctamente sin generar ningún tipo de inconveniente.

4.3.3 Pruebas técnicas

4.3.3.1 Desempeño de la app en una red Wi-Fi y en una red móvil

Esta prueba se desarrolló con la finalidad de comparar el desempeño de la aplicación en una red Wi-Fi y en una red móvil. Para el cumplimiento de esta prueba se trabajó con dos Smartphone: Samsung Galaxy S5 y un Samsung Galaxy S4 y se cronometró el tiempo de

respuesta en las cinco actividades de la aplicación móvil, durante su ejecución en tres lugares diferentes de la Ciudad de Tulcán. Obteniendo los siguientes resultados:

Tabla 9. Desempeño de la aplicación

VERSION S.O	MARCA DE SMARTPHONE	PRUEBA	RED PRUEBA 1		RED PRUEBA 2	
			MOVIL HSPA +	WIFI SUBIDA 0,39 mbps	MOVIL HSPA +	WIFI SUBIDA 0,39 mbps
ANDROID 5.0 LOLLIPOP	SAMSUNG S5	REGISTRO USUARIO	00:00:03.004	00:00:02.880	00:00:03.448	00:00:02.729
		LOGIN	00:00:06.015	00:00:00.820	00:00:04.796	00:00:03.273
		LOGIN CHAT	00:00:01.420	00:00:01.009	00:00:03.879	00:00:00.903
		ENVIAR MENSAJE	00:00:03.286	00:00:01.772	00:00:02.081	00:00:02.178
		REGISTRAR SITIO	00:00:04.279	00:00:03.423	00:00:04.280	00:00:02.730
ANDROID 4.4.2 JELLY BEAN	SAMSUNG S4	REGISTRO USUARIO	00:00:03.059	00:00:02.646	00:00:03.575	00:00:02.570
		LOGIN	00:00:03.554	00:00:00.891	00:00:04.296	00:00:01.615
		LOGIN CHAT	00:00:01.899	00:00:01.507	00:00:02.002	00:00:02.682
		ENVIAR MENSAJE	00:00:02.815	00:00:04.230	00:00:02.918	00:00:03.439
		REGISTRAR SITIO	00:00:04.296	00:00:02.854	00:00:04.548	00:00:02.997

Elaborado por: El Autor

Con los resultados obtenidos en la prueba realizada se puede señalar que la utilización de la red Wi-Fi abierta que se encontraba en cada uno de los lugares visitados, proporcionó una intensidad de la señal muy buena. Mientras que la red móvil que se utilizó fue de la operadora “Movistar”, conectándose el dispositivo móvil a una red HSPA +.

En esta prueba se concluye que los tiempos obtenidos en una red Wi-Fi abierta y en una red móvil son relativamente bajos, sin embargo, cabe señalar que:

- La red Wi-Fi permite una limitada movilización ya que no se encuentra disponible en todos los lugares donde se desea levantar la información en campo.

- La red móvil, permite una gran movilidad y disponibilidad.
- Los tiempos pueden variar según la disponibilidad de las redes y de acuerdo al lugar donde se encuentre el usuario.
- La velocidad del tiempo de respuesta depende relativamente del número de app que estén instaladas en el dispositivo y de la disponibilidad de conexión y consumo de memoria que la app en desarrollo tenga.

4.3.3.2 Comprobación del funcionamiento de aplicación móvil en diferentes versiones de Android

Esta prueba se realizó con el objetivo de comprobar el funcionamiento de aplicación móvil en varios dispositivos móviles, con diferentes versiones del sistema operativo Android, con lo cuál se obtuvo los siguientes resultados:

Tabla 10. Funcionamiento de aplicación móvil en diferentes versiones de Android

Tipo	Marca	Versión del Sistema Operativo Android	Resultados
Smartphone	Samsum GAlaxy S5	Lollipop 5.1	Funciona
Smartphone	Sony Xpiria Z3	Lollipop 5.1	Funciona
Smartphone	Samsum Galaxy S4	Jelly Bean 4.4.2	Funciona
Tablet	Acer Iconia B1-710	Jelly Bean 4.12	Funciona

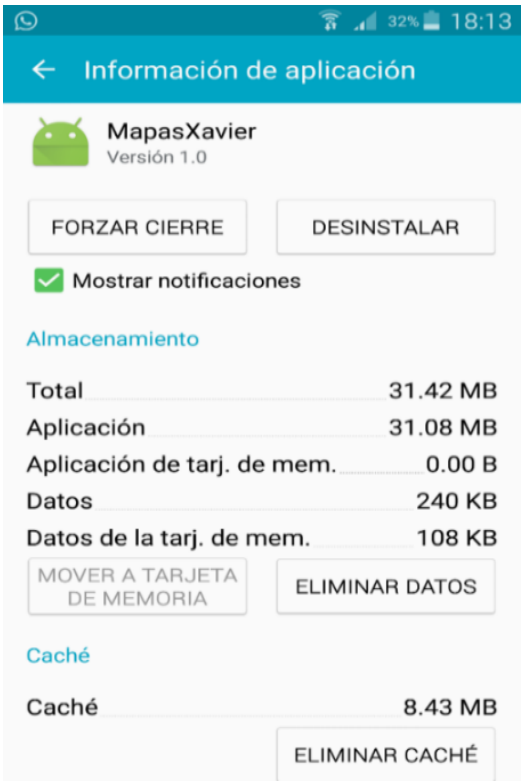
Elaborado por: El Autor

De acuerdo a los resultados obtenidos de la rueba realizada se concluye que la aplicación móvil es compatible con diferentes versiones del sistema operativo Android en varios dispositivos móviles, corooborado el planteamiento realizado en el diseño de la misma

4.3.3.3 Consumo de espacio

La aplicación tiene un tamaño de 9.1 megas en el apk, e instalado pesa 31.42 megas como se puede ver a continuación.

Ilustración 34. Consumo de espacio de la aplicación

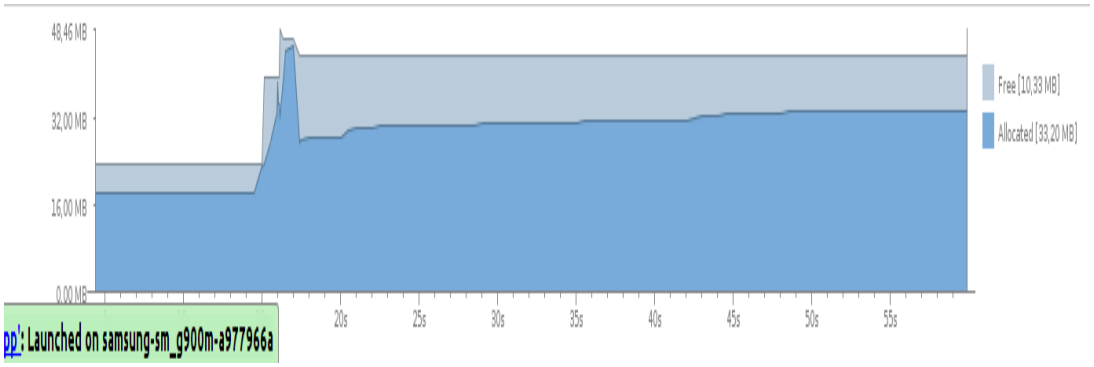


Elaborado por: El Autor

4.3.3.4 Consumo de memoria

Como se había mencionado con anterioridad se va a utilizar el profiler del Android Studio para medir el consumo de memoria de la aplicación. El consumo de memoria va desde los 16 mb cuando la aplicación arranca hasta los 50 mb cuanto todos los componentes se han cargado como se muestra en el siguiente gráfico.

Ilustración 35. Consumo de memoria



Elaborado por: El Autor

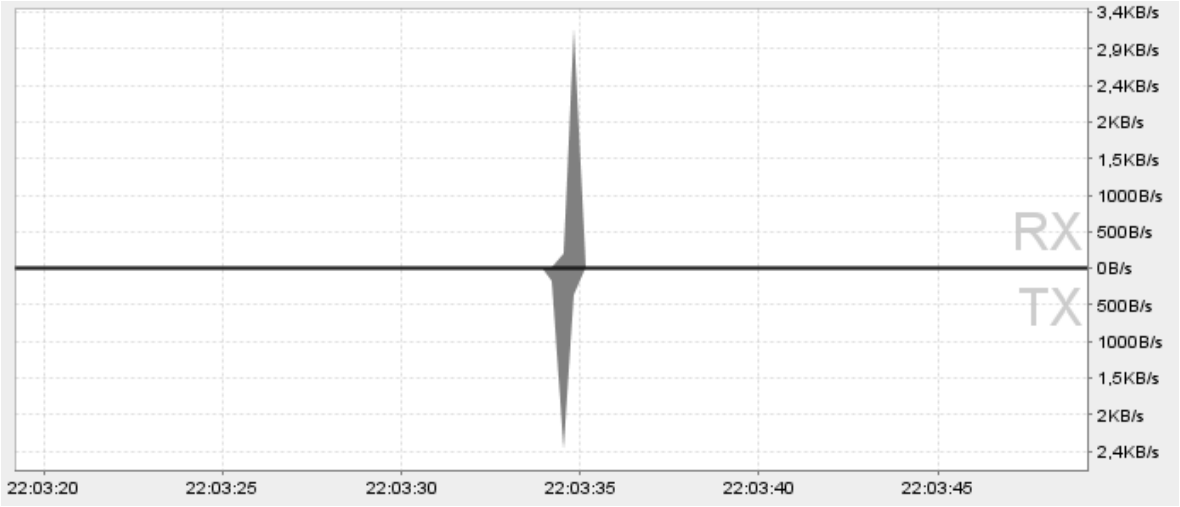
4.3.3.5 Consumo de datos por Funcionamiento

Para la siguiente prueba se utilizó Android Device Monitor que es una herramienta integrada a la plataforma Android Studio, la cual proporcionó elementos que sirvieron para monitorear diferentes dispositivos Android y con ello poder observar su comportamiento en lo que a conectividad se refiere, en tal virtud se plateó pruebas con la finalidad de conocer el consumo de datos en los diferentes módulos de funcionamiento de la aplicación móvil tal como se muestra a continuación:

- **Módulo Registro de usuario**

Como se aprecia en la figura al momento de registrar un usuario existe el tráfico de subida de: 2,92 kb/s y de bajada: 2.35 kb/s, lo cual permite evidenciar que para este módulo al momento de establecer la conexión y envío de información a la base de datos el requerimiento de ancho de banda en mínimo debido a que el paquete de datos enviada es pequeño.

Ilustración 36. Medición Consumo de datos Registro de Usuarios

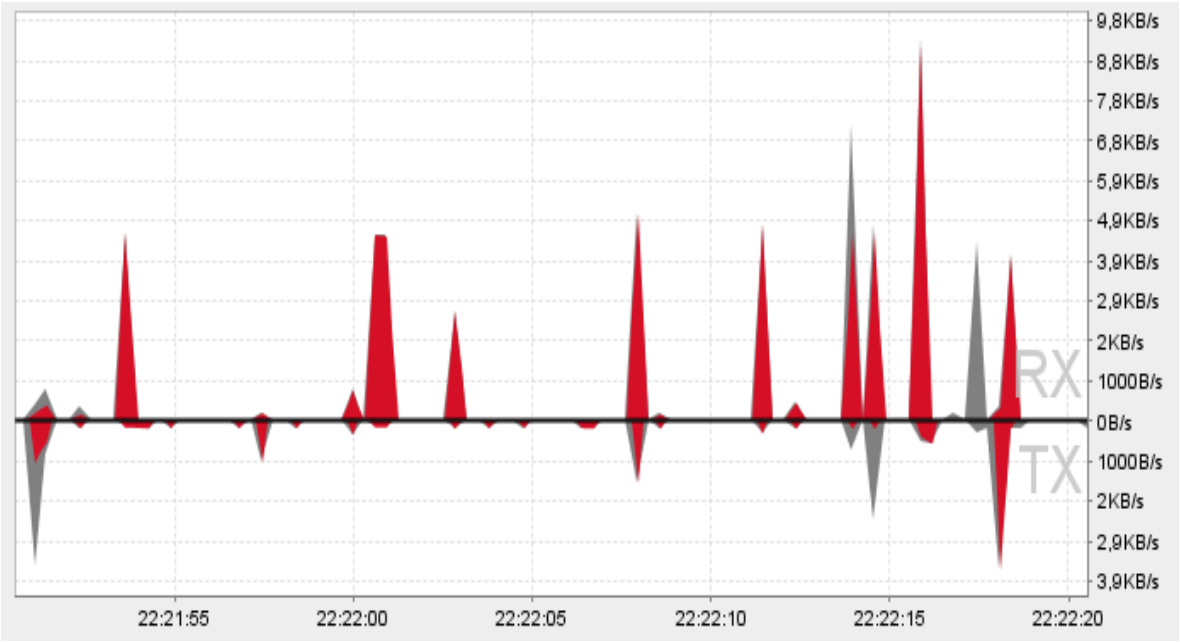


Elaborado por: El Autor

- **Módulo Login (ingreso a la aplicación)**

Al momento que se ingresa a la aplicación se llega a aproximadamente a 1000 b/s en información de subida, y de bajada hasta los 3.5 kb/s, tal como se muestra a continuación.

Ilustración 37. Medición Consumo de datos Login



Elaborado por: El Autor

- **Módulo Sinch Chat**

El Módulo de Chat se divide en dos componetes mismos que serán analizados de la siguiente manera:

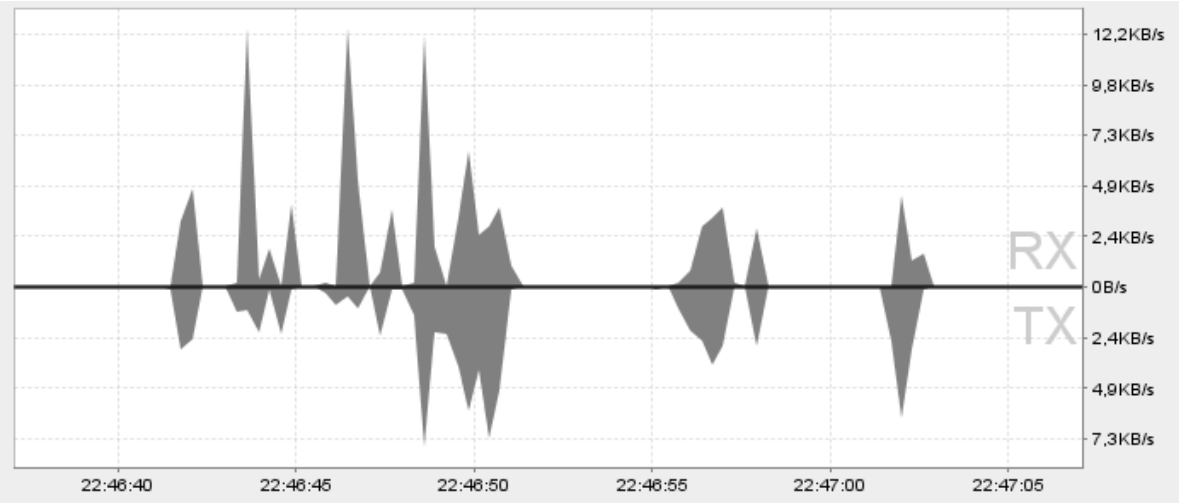
Login para utilizar el servicio de chat de la aplicación

- ✓ Para inicio del servicio de Sinch chat se establece un consumo de 4,9 kbps de subida y de bajada 2.6 kbps.

Envío y recepción de mensajes

- ✓ El consumo de ancho de banda para el envío de mensajes alcanza picos de 12,2 kbps mientras que para la recepción de información del mensaje fue de 7,3 kbps siendo este el servicio de mayor consumo de ancho de banda.

Ilustración 38. Medición de datos Chat



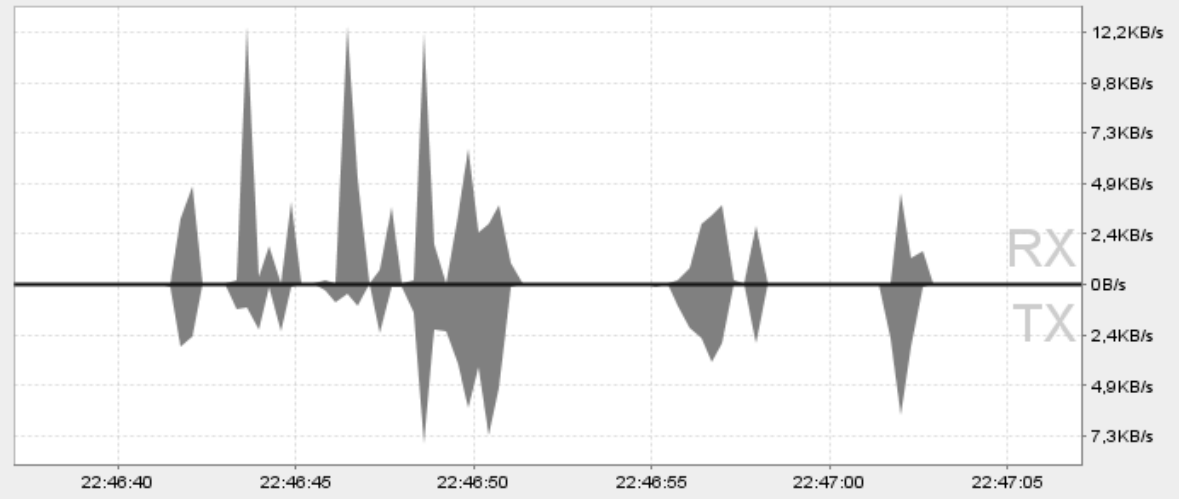
Elaborado por: El Autor

- **Módulo de Navegación en mapas**

Por medio de la actualización de las capas de Google Maps, se conoce que el consumo de datos de subida 12,2 kbps entre sus picos más altos y de 7,3 kbps de bajada, siendo este

uno de los servicios que más ancho de datos consume, debido a que las capas de los mapas e imágenes precisan más datos para cumplir con el proceso de visualización.

Ilustración 39. Medición de datos navegación de Mapas



Elaborado por: El Autor

- **Módulo Registro de formulario**

Mediante el registro de formulario se envía la información levantada a la base de datos, utilizando 4,1 kbps para la subida y 2kbps de bajada.

Ilustración 40. Medición de datos Formulario

Elaborado por: El Autor

Resumen de pruebas de consumo de datos

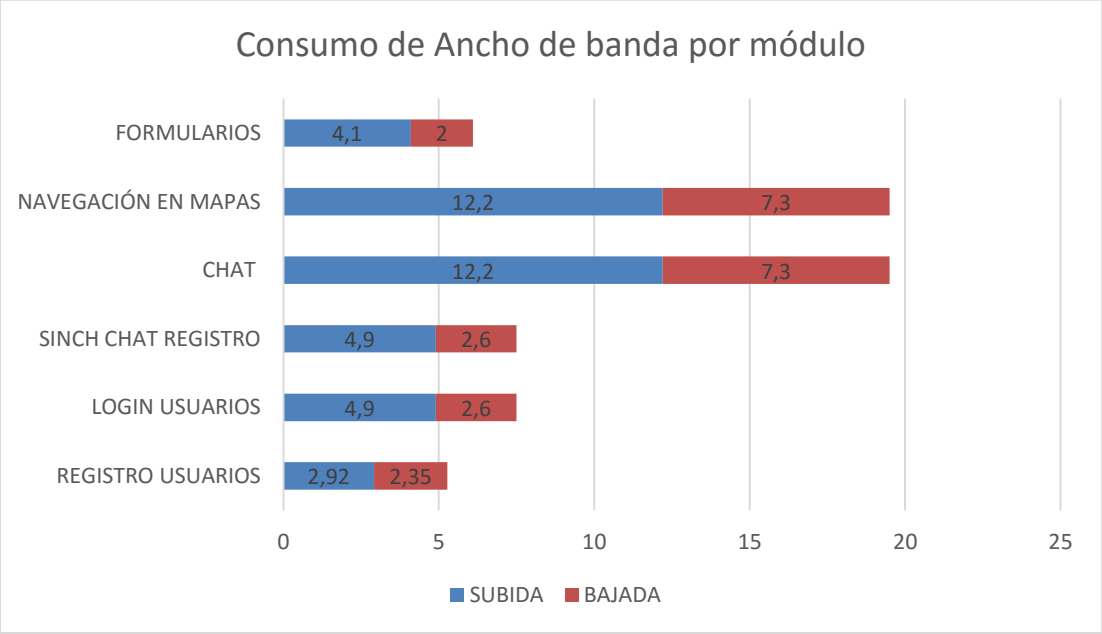
En la tabla siguiente e realiza un análisis comparativo sobre los diferentes consumos de datos que requieren los módulos de la app en su funcionamiento, pudiéndose observar que el registro de usuarios presenta el menor consumo de datos mientras que el chat y la navegación en mapas presentar el consumo más alto de toda la app debido a que estos dos módulos se conectan a servidores externos.

Tabla 11. Pruebas de Consumo de datos

MÓDULO	**SUBIDA**	**BAJADA**	
REGISTRO USUARIOS	2,92	2,35	KBPS
LOGIN USUARIOS	4,9	2,6	KBPS
SINCH CHAT REGISTRO	4,9	2,6	KBPS
CHAT	12,2	7,3	KBPS
NAVEGACIÓN EN MAPAS	12,2	7,3	KBPS
FORMULARIOS	4,1	2	KBPS
PROMEDIO DE NAVEGACION	**6,87**	**4,025**	**KBPS**

Elaborado por: El Autor

Ilustración 41. Consumo de ancho de banda por mòdulo



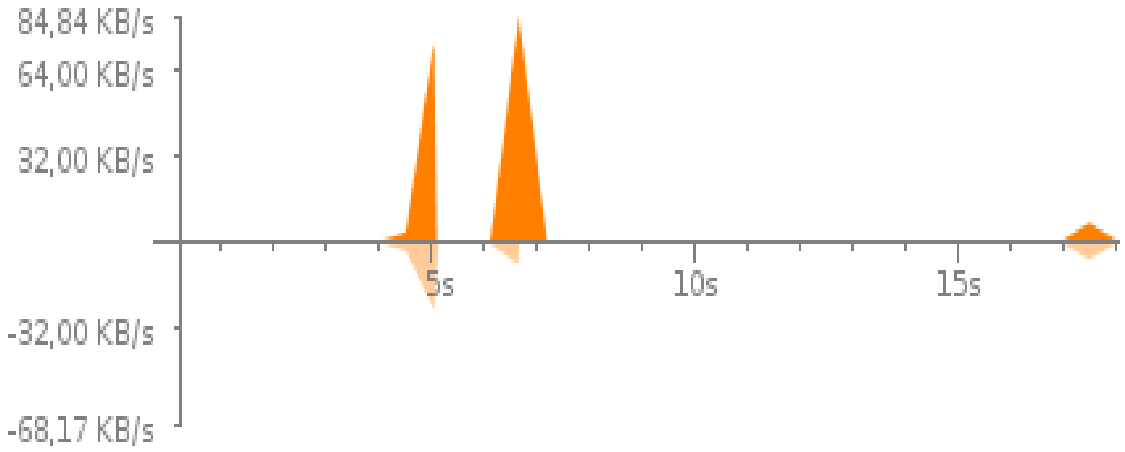
Elaborado por: El Autor

4.3.3.6 Consumo de datos por posicionamiento

El consumo de datos de lo va a realizar bajo los mismos parámetros de revisión del caso de la memoria.

La transmisión de datos se encuentra en un rango que va desde los 8 kbps hasta los 83 kbps, la mayor medición como en el caso anterior se da al inicio de la aplicación y el resto del tiempo se mantiene una constante de 8kbps por posicionamiento cada 40 segundos, adicionalmente se produce una carga de 10 kbps por envío de formulario levantado.

Ilustración 42. Consumo de datos



Elaborado por: El Autor

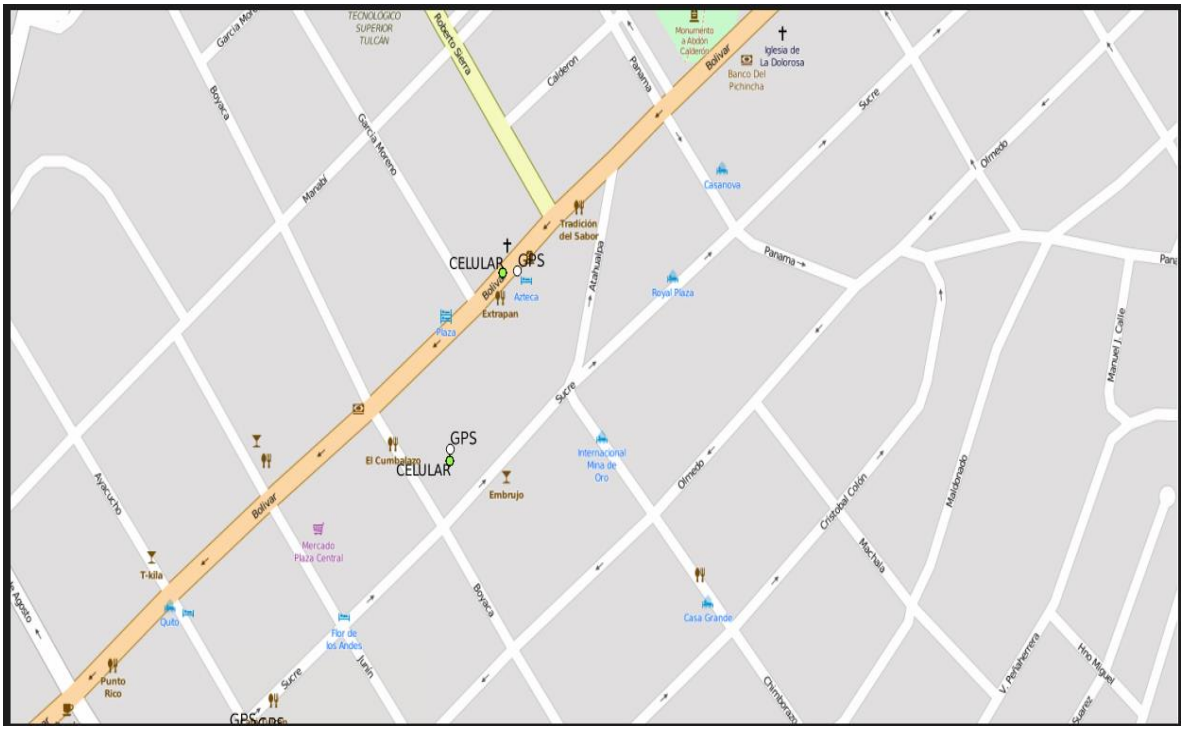
4.3.3.7 Prueba de posicionamiento de los usuarios

Para el desarrollo de esta prueba se trabajó con diez usuarios mismos que se ubicaron en diferentes sitios de la ciudad de Tulcán con el fin de georeferenciar: la posición de usuarios, fecha y hora de posición, en la que cada uno de ellos se encontraba utilizando para ello un equipo manual GPS y un Smartphone con la aplicación móvil previamente instalada.

Ilustración 43. Posicionamiento de los usuarios utilizando la aplicación móvil

Elaborado por: El Autor

Ilustración 44. Posicionamiento de los usuarios utilizando un GPS manual



Elaborado por: El Autor

Ilustración 45. Distancias de puntos tomados con GPS y celular

Elaborado por: El Autor

Los puntos georeferenciados que se observan en la ilustración 45. indican el siguiente resultado:

- Para el punto 1 la diferencia en metros que hay entre el GPS de precisión y el GPS del Smartphone es de 10,653 metros de distancia.
- Para el punto 2 la diferencia que hay entre el GPS de precisión y el Smartphone es de 6,732 metros de distnacia.

Con los dos puntos antes mencionados se puede señalar que el margen de error existente entre los dos equipos utilizdos para la toma de coordenadas es mínimo por lo tanto no varía en gran escala la posición de un sitio.

Una vez realizadas las pruebas con los dos equipos se concluye que:

Aplicación Móvil

- Por medio del uso de la aplicación móvil en un Smartphone se puede controlar la posición de cada usuario, asi como evidenciar la fecha y hora de la visita realizada a determinado sitio a través de la aplicación web en tiempo real.
- El posicionamiento de los usuarios utilizando la aplicación móvil se georeferencia automáticamente y se evidencia a través de la librería de Google Maps de la aplicación.
- Con el uso de la aplicación se determina con menor exactitud las coordenadas sobre la ubicación de sitios turísticos que de un GPS de presición.
- La velocidad en el levantamiento de coordenadas por medio del uso de la aplicación es inmediata desde el instante que se activa el servicio de GPS en el Smartphone
- Los datos levantados in situ por medio del uso de la aplicación se envían automáticamente al servidor donde se almacenan.
- Los datos levantados con la aplicación se pueden observar automáticamente georeferenciados en el sistema web instalado en el servidor.

Equipo GPS manual

- Por medio del uso del equipo GPS de presición únicamente se puede tomar las coordenadas de los sitios visitados, mas no se puede evidenciar el posicionamiento de los usuarios in situ y en tiempo real.
- El posicionamiento de los usuarios utilizando el GPS de presición se georeferencia por medio de un gestor GIS luego de cumplir un proceso de transformación de coordenadas geográficas para que sean visualizados tanto en plataformas libres como Google Maps, Open Street maps, entre otras o a través de mapas desarrollados para los fines pertinentes.
- Con el uso del GPS de precisión se determina con mayor exactitud las coordenadas sobre la ubicación de sitios turísticos.

- La velocidad en el levantamiento de coordenadas por medio del uso del GPS de precisión es menor ya que necesita esperar que el GPS logre sincronizarse con los satélites disponibles para poder levantar los puntos con exactitud.
- Los datos levantados in situ por medio del uso del GPS deben ser procesardos en un gestor GIS previo al ingreso al a la base de datos donde deben ser almacenados.

CAPITULO V
CONLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

- Entre las distintas plataformas y sistemas operativos que utilizan los dispositivos móviles, Android es quien se encuentra entre los mejores por ser un sistema operativo libre (open sourse), estable y con mayor soporte, tanto para el entorno de usuario como de desarrollo, dando un sinnúmero de posibilidades y utilitarios que permitirán mejorar la experiencia de navegación y utilización de smartphones., además se facilitó de alguna manera el desarrollo en Android ya que este emplea lenguaje Java con el cual se está familiarizado.

- Las redes GPS son de uso gratuito otorgado por el gobierno de los EE. UU que han permitido grandes cambios en la sociedad. Las aplicaciones GPS crecen cada día de manera vertiginosa y se están volviendo cotidianas en la vida y su sinnúmero de servicios que ofrece, han hecho una herramienta fundamental en diferentes campos: seguridad, movilidad, transporte, entre otras.

- La usabilidad de la aplicación es realmente intuitiva, permitiendo al usuario poder navegar por la aplicación y todos sus componentes sin necesidad de recurrir a capacitaciones extensas y/o conocimientos especializados en el tema, es necesario saber de los funcionamientos del Smartphone, herramientas de geoposicionamiento y chat para adaptarse pronto a la aplicación.

- Para el funcionamiento de la aplicación móvil es necesario contar con conexión a internet que puede ser mediante una red de datos en cualquiera de las operadoras móviles que existen en el país o apoyarse en las conexiones WI-FI que tienen los diferentes destinos turísticos que cuentan con el servicio, ya que por medio del internet se puede enviar y recibir datos desde y hacia los servidores, sin embrago se debe tomar en cuenta que los tiempos pueden variar según la disponibilidad de las redes y de acuerdo al lugar donde se encuentre el usuario, así como la velocidad del tiempo de respuesta depende relativamente del número de aplicaciones que

estén instaladas en el dispositivo y de la disponibilidad de conexión y consumo de memoria que la aplicación en desarrollo tenga.

- La aplicación hace emisión de pequeños paquetes de datos que a su vez no provocan un mayor consumo de ancho de banda y memoria del dispositivo en sus cuatro módulos: Registro-login, Chat, navegación en google maps y levantamiento de información permitiendo al usuario poder obtener datos inmediatos de posicionamiento e interacción en tiempo real con otros usuarios conectados.

5.2 RECOMENDACIONES

- La constante evolución de dispositivos Smartphone hace que las aplicaciones Android sean cada vez más adaptables a las tecnologías, para ello se deben respetar los estándares de desarrollo para que el proceso evolutivo de los dispositivos móviles vaya a la par con las app's y así optimizar tiempo, recursos y dinero empleado en desarrollo de aplicaciones.

- Mejorar los planes de las librerías tanto para Sinch como para Google maps permitirán que las aplicaciones desarrolladas con estas api's puedan mejorar la experiencia de manejo para el usuario.

- Uno de los elementos que mayor consumo de batería realiza es el GPS, por ello se recomienda realizar la activación cuando se vaya a realizar trabajo de campo o cuando se lo amerite ya que el dispositivo permite enviar posicionamientos cada 40 segundos en la aplicación sin contar aplicaciones que necesariamente lo activan constantemente.

- Gestionar con la empresa Google para mejorar el posicionamiento de algunas zonas de la provincia del Carchi en la plataforma Google maps ya que el margen de

error es demasiado amplio de acuerdo a Sistemas de información geográficos desarrollados en la Prefectura del Carchi.

- Para la emulación se recomienda usar un dispositivo real ya sea una Tablet o un Smarthphone ya que el ADV o Emulador de Android carece de algunas herramientas como GPS, Brújula, etc y sobre todo evitar consumir recursos de procesamiento del computador y proder probar la verdadera funcionalidad de la App en un entorno real.

- Por lo expuesto y en referencia al caso de estudio se debe actualizar constantemente la información en la web que en este caso sería la información turística, y desarrollar herramientas que permitan la automatización inmediata de la información utilizando los dispositivos de uso diario para optimizar temas de inversión y tiempo.

REFERENCIAS BIBLIOGRAFICAS

[1] Superintendencia de Telecomunicaciones, (2012) “Evolución de la Telefonía Móvil en el Ecuador” Revista Institucional SUPERTEL nº16, pp. 4, 31,32.

[2] Ángel J. Vico, (2011) "Arquitectura de Android" Recuperado de http://androideity.com/2011/07/04/arquitectura-de-android/

[3] Jose Angel Zamora, (2014) "Aprende Android en 20 conceptos: Conceptos 1 y 2" Recuperado de http://www.elandroidelibre.com/2014/02/aprende-android-en-20-conceptos-conceptos-1-y-2.html

[4] IDE, NIVEL BÁSICO, (2014) "IDE: Entornos Integrados de Desarrollo para Android" Recuperado de http://academiaandroid.com/ide-entornos-integrados-de-desarrollo-para-android/

[5] Libros Web, (2015) "Google Maps" Recuperado de http://librosweb.es/libro/ajax/capitulo_9/google_maps.html

[6] Julián Pérez Porto y Ana Gardey, (2013) "Definicion de WiFi-Qué es, Significado y Concepto" Recuperado de http://definicion.de/wifi/#ixzz48B9N2kvD

[7] GPS.gov, Official U.S. Government information about the Global Positioning System (GPS) and related topics , (2016) "The Global Positioning System" Recuperado de http://www.gps.gov/systems/gps/spanish.php

[8] Wikipedia, (2016) "Simple Object Access Protocol" https://es.wikipedia.org/wiki/Simple_Object_Access_Protocol

ANEXOS

ANEXO 1: CONFIGURACIÓN DEL MAPA GOOGLE MAPS

```
//fijar centro en ecuador

private LatLngBounds ECUADOR = new LatLngBounds(
new LatLng(-1.17, -81.30), new LatLng(1.01, -75.41));
public void setupMap() {
// habilitar controles de zoom
mMap.getUiSettings().setZoomControlsEnabled(true);
mMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
//habilitar brújula
mMap.getUiSettings().setCompassEnabled(true);
mMap.setMyLocationEnabled(true);
mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(ECUADOR.getCenter(), 8));
}
@Override
public void onMapReady(GoogleMap googleMap) {
mMap = googleMap;
  setupMap();
   }
googleMap.clear();
}
```

ANEXO 2. OBTENIENDO LA UBICACIÓN DEL USUARIO

```
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

- A continuación se debe conectar a los servicios de Google Play API, esto se lo realiza a través de la configuración de una variable de tipo GoogleAPIClient, la cual devolverá la conexión al API. que se debe conectar en el evento onStart() y desconectar en el evento onStop de la aplicación.

```
protected synchronized void buildGoogleApiClient() {
mGoogleApiClient = new GoogleApiClient.Builder(this.getActivity())
        .addConnectionCallbacks(this)
        .addOnConnectionFailedListener(this)
        .addApi(LocationServices.API)
        .build();
  createLocationRequest();
}
@Override
public void onStop() {
mGoogleApiClient.disconnect();
super.onStop();
}
```

- Una vez conectado a Google Play Services, se puede obtener la ubicación expresada en latitud y longitud para este caso usando los métodos **getlatitude()** y **getLongitude()** del objeto **mLastLocation** como se muestra a continuación.

```
Location mLastLocation;
@Override
public void onLocationChanged(Location location) {
mLastLocation = location;
double dLatitude = mLastLocation.getLatitude();
double dLongitude = mLastLocation.getLongitude();
```

}

- Definir un objeto LocationRequest, el cual va a determinar las características de precisión y tiempo de respuesta, a través de sus distintos parámetros como son el intervalo a través del método **setInverval(),** la velocidad más rápida en la que se recibe información a través de **setFastestInterval()**, los dos expresados en milisegundos, el segundo tiene relación con el uso de la ubicación que hacen otras aplicaciones. La prioridad se la revisa a través del metódo **setPriority()**, que para el caso actual se ha fijado en **PRIORITY_HIGH_ACCURACY** , con el que se obtiene la precisión más precisa posible, que es equivalente a usar un GPS para determinar la posición, a continuación se muestran las paramétros configurados.

```
LocationRequest mLocationRequest;
//location request
// actualiza cada 40 segundos min 20 segundos
protected void createLocationRequest() {
mLocationRequest = new LocationRequest();
mLocationRequest.setInterval(40000);
mLocationRequest.setFastestInterval(20000);
//prioridad alta precision
mLocationRequest.setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
}
```

ANEXO 3. INTEGRANDO CHAT A LA APLICACIÓN

- Registrarse una cuenta en sinch y parse para poder tener acceso a los servicios, en las direcciones https://www.sinch.com y https://www.parse.com/

- Se deben habilitar el logueo del usuario contra Parse, para esto se deben habilitar los permisos en el **AndroidManifest.xml.** Adicionalmente se deben configurar las dependencias en Gradle en el archivo **build.gradle** en la sección de dependencias.

AndroidManifest.xml

```
<uses-permissionandroid:name="android.permission.INTERNET"/>
<uses-permissionandroid:name="android.permission.ACCESS_NETWORK_STATE"/>
```

build.gradle

```
compile files('libs/Parse-1.9.3.jar')
compile files('libs/bolts-android-1.2.0.jar')
compile files('libs/ParseCrashReporting-1.9.3.jar')
compile files('libs/sinch-android-rtc-3.6.2.jar')
compile files('libs/ParseFacebookUtilsV3-1.9.3.jar')
compile files('libs/ksoap2-android-assembly-2.4-jar-with-dependencies.jar')
```

- Habilitados los permisos necesarios se deben crear una clase que extienda la clase Application en la cual en el evento OnCreate se va a inicializar el acceso a Parse con el usuario y contraseña que se ha previsto en la creación de la cuenta en este BAAS.

```
public class MapasXavo extends Application {
@Override
public void onCreate() {
super.onCreate();
Log.v("Inicializando", "cargando perfil");
Parse.initialize(this, "app-id", "client-key");
}
}
```

Ahora se debe generar la interfaz requerida para loguearse o crear un nuevo usuario dentro del chat, la interfaz generada tiene dos botónes una para loguearse y el otro para iniciar

sesión, además de dos casillas para ingresar el usuario y contraseña, esto se valida en el evento onCreateView de la clase ChatActivity, como se muestra a continuación:

1. Si el usuario accede de una manera correcta va a lanzarse la siguiente actividad llamada **ListUserActivity**, la cual contiene un **ListView** que es donde se van a cargar los usuarios generados por la aplicación, la estructura del Layout es la siguiente:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
android:paddingBottom="@dimen/activity_vertical_margin"
tools:context="com.example.paul.mapasxavo.ListUsersActivity">

<Button
style="@style/botones"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="@string/logout"
android:id="@+id/logoutButton"
android:gravity="center_vertical|center_horizontal"
android:layout_gravity="center_horizontal"
android:textSize="24sp"
android:padding="15dp"
/>
```

```
<ListView xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:padding="16dp"
android:background="@color/colorPrimary"
android:id="@+id/usersListView">
</ListView>
</LinearLayout>
```

Este layout necesita ser cargado por los usuarios que se han creado en Parse para usar con la aplicación, los cuales se deben consultar en el la clase ListUserActivity dentro de un método llamado setConversationList(), como se muestra a continuación.

```
//display clickable a list of all users
private void setConversationsList() {
currentUserId = ParseUser.getCurrentUser().getObjectId();
names = new ArrayList<String>();
  ParseQuery<ParseUser> query = ParseUser.getQuery();
  query.whereNotEqualTo("objectId", currentUserId);
  query.findInBackground(new FindCallback<ParseUser>() {
public void done(List<ParseUser> userList, com.parse.ParseException e) {
if (e == null) {
for (int i = 0; i < userList.size(); i++) {
names.add(userList.get(i).getUsername().toString());
          }
usersListView = (ListView) rootView.findViewById(R.id.usersListView);
namesArrayAdapter =
new ArrayAdapter<String>(getActivity(),
                    R.layout.user_list_item, names);
```

```
usersListView.setAdapter(namesArrayAdapter);
usersListView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
@Override
public void onItemClick(AdapterView<?> a, View v, int i, long l) {
                openConversation(names, i);
            }
          });
        } else {
          Toast.makeText(getActivity(),
"Error loading user list",
                Toast.LENGTH_LONG).show();
        }
    }
  });
}
@Override
public void onResume() {
  setConversationsList();
super.onResume();
}
```

Adicionalmente se genera un método para abrir la conversación con la persona que se desea esto se lo realiza con openConversation() de la siguiente manera:

```
//open a conversation with one person
public void openConversation(ArrayList<String> names, int pos) {
  ParseQuery<ParseUser> query = ParseUser.getQuery();
  query.whereEqualTo("username", names.get(pos));
  query.findInBackground(new FindCallback<ParseUser>() {
public void done(List<ParseUser> user, com.parse.ParseException e) {
```

```
if (e == null) {
            Intent intent = new Intent(getActivity(), MensajeriaActivity.class);
            intent.putExtra("RECIPIENT_ID", user.get(0).getObjectId());
            startActivity(intent);
        } else {
            Toast.makeText(getActivity(),
                    getString(R.string.error_find_user),
Toast.LENGTH_SHORT).show();
        }
    }
  });
}
```

Realizado este proceso se debe arrancar el cliente de Sinch, para esto se deben configurar algunos permisos que son necesarios dentro de la aplicación, esto se lo realiza a través de **AndroidManifest.xml** como se ve a continuación:

```
<uses-permissionandroid:name="android.permission.INTERNET"/>
<uses-permissionandroid:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permissionandroid:name="android.permission.RECORD_AUDIO"/>
<uses-permissionandroid:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
<uses-permissionandroid:name="android.permission.READ_PHONE_STATE"/>
```

- A continuación para escuchar las peticiones de los usuarios se debe crear un servicio donde se arrancará el servicio Sinch de cliente cuando un usuario se loguee. Una vez logueado el usuario correrá en backgroud mientras la aplicación está abierta y será usado para enviar y recibir mensajes. El método en el que se dispara el evento es el onStartCommand, haciendo referencia a **startSinchClient** como se muestra a continuación.

```
@Override
```

```
public int onStartCommand(Intent intent, int flags, int startId) {
currentUserId = ParseUser.getCurrentUser().getObjectId();
if (currentUserId != null && !isSinchClientStarted()) {
     startSinchClient(currentUserId);
  }
broadcaster = LocalBroadcastManager.getInstance(this);
return super.onStartCommand(intent, flags, startId);
}
public void startSinchClient(String username) {
sinchClient =
Sinch.getSinchClientBuilder().context(this).userId(username).applicationKey(APP_KEY)
        .applicationSecret(APP_SECRET).environmentHost(ENVIRONMENT).build();
sinchClient.addSinchClientListener(this);
sinchClient.setSupportMessaging(true);
sinchClient.setSupportActiveConnectionInBackground(true);
sinchClient.checkManifest();
sinchClient.start();
}
```

- Verificar que el cliente Sinch se haya inicializado, debido a que por problemas de conectividad puede tardar en arrancar el servicio del cliente se por este motivo se crea un spinner wheel que se va a mostrar mientras el servicio arranca como se muestra a continuación en la clase **MessageService.java** y en la clase **ListUsersActivity.java**

MessageService.java

```
@Override
public int onStartCommand(Intent intent, int flags, int startId) {
 broadcaster = LocalBroadcastManager.getInstance(this);
 return super.onStartCommand(intent, flags, startId);
}
```

```
@Override
public void onClientStarted(SinchClient client) {
  broadcastIntent.putExtra("success", true);
  broadcaster.sendBroadcast(broadcastIntent);
}
@Override
public void onClientFailed(SinchClient client, SinchError error) {
  broadcastIntent.putExtra("success", false);
  broadcaster.sendBroadcast(broadcastIntent);
  sinchClient = null;
}
```

ListUsersActivity.java

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
   rootView = inflater.inflate(R.layout.activity_list_users, container, false);
  showSpinner();
 logoutButton = (Button) rootView.findViewById(R.id.logoutButton);
  logoutButton.setOnClickListener(new View.OnClickListener() {
     @Override
     public void onClick(View view) {
        getActivity().stopService(new Intent(getActivity(), MessageService.class));
        ParseUser.logOut();
        FragmentManager fragmentManager = myContext.getSupportFragmentManager();
        //start transaction
        FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
        ChatActivity chatActivity = new ChatActivity();
        fragmentTransaction.addToBackStack("xyz");
        fragmentTransaction.hide(ListUsersActivity.this);
```

```
            fragmentTransaction.commit();
        }
    });
    return rootView;
}
//show a loading spinner while the sinch client starts
private void showSpinner() {
    progressDialog = new ProgressDialog(getActivity());
    progressDialog.setTitle(getString(R.string.progress_dialog));
    progressDialog.setMessage(getString(R.string.progress_dialog_message));
    progressDialog.show();
    receiver = new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
            Boolean success = intent.getBooleanExtra("success", false);
            progressDialog.dismiss();
            if (!success) {
                Toast.makeText(getActivity(), getString(R.string.error_list_users),
Toast.LENGTH_LONG).show();
            }
        }
    };
    LocalBroadcastManager.getInstance(getActivity()).registerReceiver(receiver, new
IntentFilter("com.example.paul.mapasxavo.ListUsersActivity"));
}
```

- Arrancado el servicio ahora se debe proceder a enviar los mensajes entre los usuarios registrados, esto se realiza creando un listener en la lista de usuarios generados en la clase **ListUsersActivity.java**, y mostrado en una nueva actividad llamada **MensajeriaActivity.java**. A continuación se muestra el listener y la actividad.

ListUserActivity.java

```
//open a conversation with one person
public void openConversation(ArrayList<String> names, int pos) {
  ParseQuery<ParseUser> query = ParseUser.getQuery();
  query.whereEqualTo("username", names.get(pos));
  query.findInBackground(new FindCallback<ParseUser>() {
     public void done(List<ParseUser> user, com.parse.ParseException e) {
        if (e == null) {
           Intent intent = new Intent(getActivity(), MensajeriaActivity.class);
           intent.putExtra("RECIPIENT_ID", user.get(0).getObjectId());
           startActivity(intent);
        } else {
           Toast.makeText(getActivity(),
                 getString(R.string.error_find_user),
                 Toast.LENGTH_SHORT).show();
        }
     }
  });
}
```

MensajeriaActivity.java

```
public class MensajeriaActivity extends ActionBarActivity {
  private String recipientId;
  private EditText messageBodyField;
  private String messageBody;
  private MessageService.MessageServiceInterface messageService;
  private MessageAdapter messageAdapter;
  private ListView messagesList;
  private String currentUserId;
  private ServiceConnection serviceConnection = new MyServiceConnection();
```

```
 private MessageClientListener messageClientListener = new MyMessageClientListener();
 @Override
 protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_mensajeria);
    bindService(new Intent(this, MessageService.class), serviceConnection,
BIND_AUTO_CREATE);
    Intent intent = getIntent();
    recipientId = intent.getStringExtra("RECIPIENT_ID");
    currentUserId = ParseUser.getCurrentUser().getObjectId();
    messagesList = (ListView) findViewById(R.id.listMessages);
    messageAdapter = new MessageAdapter(this);
    messagesList.setAdapter(messageAdapter);
    populateMessageHistory();
    messageBodyField = (EditText) findViewById(R.id.messageBodyField);
    findViewById(R.id.sendButton).setOnClickListener(new View.OnClickListener() {
       @Override
       public void onClick(View view) {
          sendMessage();
       }
    });
 }
 //get previous messages from parse & display
 private void populateMessageHistory() {
    String[] userIds = {currentUserId, recipientId};
    ParseQuery<ParseObject> query = ParseQuery.getQuery("ParseMessage");
    query.whereContainedIn("senderId", Arrays.asList(userIds));
    query.whereContainedIn("recipientId", Arrays.asList(userIds));
    query.orderByAscending("createdAt");
```

```
    query.findInBackground(new FindCallback<ParseObject>() {
      @Override
      public void done(List<ParseObject> messageList, com.parse.ParseException e) {
        if (e == null) {
          for (int i = 0; i < messageList.size(); i++) {
            WritableMessage message = new
WritableMessage(messageList.get(i).get("recipientId").toString(),
messageList.get(i).get("messageText").toString());
            if (messageList.get(i).get("senderId").toString().equals(currentUserId)) {
              messageAdapter.addMessage(message,
MessageAdapter.DIRECTION_OUTGOING);
            } else {
              messageAdapter.addMessage(message,
MessageAdapter.DIRECTION_INCOMING);
            }
          }
        }
      }
    });
  }
  @Override
  public void onDestroy() {
    messageService.removeMessageClientListener(messageClientListener);
    unbindService(serviceConnection);
    super.onDestroy();
  }
  private class MyServiceConnection implements ServiceConnection {
    @Override
    public void onServiceConnected(ComponentName componentName, IBinder iBinder) {
```

```
        messageService = (MessageService.MessageServiceInterface) iBinder;
        messageService.addMessageClientListener(messageClientListener);
    }
    @Override
    public void onServiceDisconnected(ComponentName componentName) {
        messageService = null;
    }
 }
      }
```

Para evitar que se envíen mensajes en blanco se agrega una validación del mismo en la clase **MensajeriaActivity.java** que se aplica en el método onCreate, como se muestra a continuación.

```
private void sendMessage() {
  messageBody = messageBodyField.getText().toString();
  if (messageBody.isEmpty()) {
     Toast.makeText(this, "Ingrese un mensaje", Toast.LENGTH_LONG).show();
     return;
  }
     messageService.sendMessage(recipientId, messageBody);
     messageBodyField.setText("");
}
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_mensajeria);
```

```
 bindService(new Intent(this, MessageService.class), serviceConnection,
BIND_AUTO_CREATE);
 Intent intent = getIntent();
 recipientId = intent.getStringExtra("RECIPIENT_ID");
 currentUserId = ParseUser.getCurrentUser().getObjectId();
 messagesList = (ListView) findViewById(R.id.listMessages);
 messageAdapter = new MessageAdapter(this);
 messagesList.setAdapter(messageAdapter);
 populateMessageHistory();
 messageBodyField = (EditText) findViewById(R.id.messageBodyField);
 findViewById(R.id.sendButton).setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
       sendMessage();
    }
 });
}
```

Para poder gestionar los mensajes se crea una inner class llamada:

MyMessageClientListener dentro de **MensajeriaActivity.java** la cual ayuda a informar el estado de los mensajes como se muestra a continuación.

```
private class MyMessageClientListener implements MessageClientListener {
 @Override
 public void onMessageFailed(MessageClient client, Message message,
                    MessageFailureInfo failureInfo) {

        Toast.makeText(MensajeriaActivity.this, failureInfo.getSinchError().getMessage(),
Toast.LENGTH_LONG).show();
```

```
  }

  @Override
  public void onIncomingMessage(MessageClient client, Message message) {
    if (message.getSenderId().equals(recipientId)) {
      WritableMessage writableMessage = new
WritableMessage(message.getRecipientIds().get(0), message.getTextBody());
      messageAdapter.addMessage(writableMessage,
MessageAdapter.DIRECTION_INCOMING);
    }
  }
  @Override
  public void onMessageSent(MessageClient client, Message message, String recipientId) {
    final WritableMessage writableMessage = new
WritableMessage(message.getRecipientIds().get(0), message.getTextBody());
    //only add message to parse database if it doesn't already exist there
    ParseQuery<ParseObject> query = ParseQuery.getQuery("ParseMessage");
    query.whereEqualTo("sinchId", message.getMessageId());
    query.findInBackground(new FindCallback<ParseObject>() {
      @Override
      public void done(List<ParseObject> messageList, com.parse.ParseException e) {
        if (e == null) {
          if (messageList.size() == 0) {
            ParseObject parseMessage = new ParseObject("ParseMessage");
            parseMessage.put("senderId", currentUserId);
            parseMessage.put("recipientId", writableMessage.getRecipientIds().get(0));
            parseMessage.put("messageText", writableMessage.getTextBody());
            parseMessage.put("sinchId", writableMessage.getMessageId());
            parseMessage.saveInBackground();
```

```
                messageAdapter.addMessage(writableMessage,
MessageAdapter.DIRECTION_OUTGOING);
            }
          }
        }
    });
 }
```

Para desplegar los mensajes se requiere de un list adapter el cual da la apariencia de un chat común, a continuación, se presenta el **MessageAdapter.java.**

```
public class MessageAdapter extends  BaseAdapter{
  public static final int DIRECTION_INCOMING = 0;
  public static final int DIRECTION_OUTGOING = 1;
  private List<Pair<WritableMessage, Integer>>messages;
  private LayoutInflater layoutInflater;
  public MessageAdapter(Activity activity) {
     layoutInflater = activity.getLayoutInflater();
     messages = new ArrayList<Pair<WritableMessage, Integer>>();
  }
  public void addMessage(WritableMessage message, int direction) {
     messages.add(new Pair(message, direction));
     notifyDataSetChanged();
  }
  @Override
  public int getCount() {
     return messages.size();
  }
```

```
@Override
public Object getItem(int i) {
   return messages.get(i);
}
@Override
public long getItemId(int i) {
   return i;
}
@Override
public int getViewTypeCount() {
   return 2;
}
@Override
public int getItemViewType(int i) {
   return messages.get(i).second;
}
@Override
public View getView(int i, View convertView, ViewGroup viewGroup) {
   int direction = getItemViewType(i);
   if (convertView == null) {
      int res = 0;
      if (direction == DIRECTION_INCOMING) {
         res = R.layout.message_right;
      } else if (direction == DIRECTION_OUTGOING) {
         res = R.layout.message_left;
      }
      convertView = layoutInflater.inflate(res, viewGroup, false);
   }
```

```
    WritableMessage message = messages.get(i).first;
    TextView txtMessage = (TextView) convertView.findViewById(R.id.txtMessage);
    txtMessage.setText(message.getTextBody());
    return convertView;
  }
}
```

- Para lograr tener un control de los mensajes estos se almacenan en Parse en el método onMessageSent de la clase MensajeriaActivity.java como se muestra a continuación.

```
@Override
public void onMessageSent(MessageClient client, Message message, String recipientId) {
  final WritableMessage writableMessage = new
WritableMessage(message.getRecipientIds().get(0), message.getTextBody());
  //only add message to parse database if it doesn't already exist there
  ParseQuery<ParseObject> query = ParseQuery.getQuery("ParseMessage");
  query.whereEqualTo("sinchId", message.getMessageId());
  query.findInBackground(new FindCallback<ParseObject>() {
    @Override
    public void done(List<ParseObject> messageList, com.parse.ParseException e) {
      if (e == null) {
        if (messageList.size() == 0) {
          ParseObject parseMessage = new ParseObject("ParseMessage");
          parseMessage.put("senderId", currentUserId);
          parseMessage.put("recipientId", writableMessage.getRecipientIds().get(0));
          parseMessage.put("messageText", writableMessage.getTextBody());
          parseMessage.put("sinchId", writableMessage.getMessageId());
          parseMessage.saveInBackground();
```

```
            messageAdapter.addMessage(writableMessage,
MessageAdapter.DIRECTION_OUTGOING);
          }
        }
    }
 });
}
```

ANEXO 4. GENERANDO LOS SERVICIOS WEB EN EL SERVIDOR I

- Para estructurar la aplicación en POO se ha generado una clase que publicará los servicios web que se han generado para el uso de la aplicación, utilizando Adodb PHP para la gestión de conexiones con la base de datos como se muestra a continuación.

```
class App  {
  public function setLocation($lat, $lon,$user){
     global $db;
     $hoy = date("Y-m-d H:i:s");
     $sql="insert into geoposicion(fecha,usuario,geom) VALUES( "
          . "'$hoy','$user',ST_GeomFromText('POINT($lon $lat)', 4326))";
     $rs = $db->Execute($sql);
      return "datos ingresados";
  }
  public function register ($username, $password){
     global $db;
     //consultar si el usuario no esta en la base de datos
     $sql="select usuario from login where usuario='$username'";
     $rs=$db->Execute($sql);
     if ($rs->RecordCount()>0){
        return "usuario ya existe";
     }
     else {
        $sql="Insert into login  (usuario,password) values ('$username', '$password')";
        $rs=$db->Execute($sql);
        return "usuario registrado";
     }
  }
  public function login ($username, $password){
     global $db;
```

```
      //consultar si el usuario no esta en la base de datos
      $sql="select usuario from login where usuario='$username' and
password='$password'";
      $rs=$db->Execute($sql);
      if ($rs->RecordCount()>0){
            $hoy = date("Y-m-d H:i:s");
         $sql="Insert into accesos  (username,fecha) values ('$username', '$hoy')";
         $rs=$db->Execute($sql);
         return "OK";
      }
      else {
      return "usuario o password incorrecto";
      }
  }
  public function getusers ($username){
      global $db;
      //consultar si el usuario no esta en la base de datos
      $sql="select distinct on (usuario) usuario, fecha, geo_id, ST_AsGeoJSON(geom) from
geoposicion
where usuario not like '$username' order by usuario,fecha desc, geo_id;";
      $rs=$db->Execute($sql);
    $geojson = array();
      $geojson['type'] = 'FeatureCollection';
      $i = 0;
      function stripslashes_deep($value)
{
  $value = is_array($value) ?
            array_map('stripslashes_deep', $value) :
            stripslashes($value);
```

```
  return $value;
}
    if (!$rs)
       print $db->ErrorMsg();
    else
    {
       while (!$rs->EOF) {
          $geojson['features'][$i] = array("type" => "Feature", "geometry" =>
json_decode($rs->fields[3]),
             "properties" => array("usuario" => $rs->fields[0], "fecha" => $rs->fields[1], "id"
=> $rs->fields[2]));
          $rs->MoveNext();
          $i++;
       }
    $rs->Close(); # optional
    #
    }
    return json_encode($geojson);
}
}
```

- La clase generada ahora debe ser publicada para que sea consumida por la aplicación, esto se lo realiza a través del siguiente código en la clase.

```
$server = new soap_server();
$server->configureWSDL("appservice", "http://appserver/webservice");
$server->register("app.setLocation",
  array("lat" => "xsd:string", "lon"=> "xsd:string", "user"=>"xsd:string"),
  array("return" => "xsd:string"),
  "http://appserver/webservice",
```

```
  "http://appserver/webservice#setPos",
  "rpc",
  "encoded",
  "set pos");
$server->register("app.register",
  array("username" => "xsd:string", "password"=> "xsd:string"),
  array("return" => "xsd:string"),
  "http://appserver/webservice",
  "http://appserver/webservice#register",
  "rpc",
  "encoded",
  "set pos");
$server->register("app.login",
  array("username" => "xsd:string", "password"=> "xsd:string"),
  array("return" => "xsd:string"),
  "http://appserver/webservice",
  "http://appserver/webservice#login",
  "rpc",
  "encoded",
  "set pos");
$server->register("app.getusers",
  array("username" => "xsd:string"),
  array("return" => "xsd:string"),
  "http://appserver/webservice",
  "http://appserver/webservice#getusers",
  "rpc",
  "encoded",
  "set pos");
@$server->service($HTTP_RAW_POST_DATA);
```

- Realizado esto se tiene listo el servidor web de la aplicación, el mismo proceso se lo va a realizar en la aplicación de turismo, como se muestra a continuación.

```
class Servicios {
  function crearTurismo($canton, $tipo, $tabla, $nombre, $direccion, $latitud, $longitud, $url) {
    if ($tabla == 'Turismo') {
       $tabla = 'geoturismo';
    }
    //codificar las variables UTF-8
    $canton=  utf8_encode($canton);
    $tipo=  utf8_encode($tipo);
    $tabla=  utf8_encode($tabla);
    $nombre=  utf8_encode($nombre);
    $direccion= utf8_encode($direccion);
    $turismo = new Turismo();
    $punto = new loadDb();
    $id = $punto->consultaId($tabla);
     $turismo->setDpa_prov("04");
    $idCanton=$punto->consultaCantonId($canton);
    $tipoP=$punto->consultaIdTipo($tipo,$tabla);
    $turismo->setDpa_canton($idCanton);
    $turismo->setDpa_parr('000000');
    $turismo->setTipo($tipoP);
    $turismo->setId($id);
    $turismo->setNombreT(strtoupper($nombre));
    $turismo->setDireccionT($direccion);
    $turismo->setLatitud($latitud);
    $turismo->setLongitud($longitud);
```

```
        $turismo->setUrl($url);
        $mensaje=$punto->cargaPunto($turismo, $tabla);
        return $mensaje;
    }
    public function crearAlojamiento
($tabla,$nroFol,$establecimiento,$propietario,$direccion,$habitaciones,$plaza,
            $precio, $mesas, $plazasComedor, $personal, $telefono, $enlace, $canton,
$tipo,$categoria,
            $lat, $lon){
        //codificar las entradas
        $tabla=  utf8_encode($tabla);
        $nroFol=  utf8_encode($nroFol);
        $establecimiento= strtoupper(utf8_encode($establecimiento));
        $propietario= strtoupper(utf8_encode($propietario));
        $direccion= strtoupper(utf8_encode($direccion));
        $canton= utf8_encode($canton);
        $categoria= utf8_encode($categoria);
        $tipo=  utf8_encode($tipo);
        $punto = new loadDb();
        if ($tabla == 'Alojamiento') {
            $tabla = 'geoalojamiento';
        }elseif ($tabla=='Recreación') {
            $tabla = 'georecreacion';
        }elseif ($tabla=='Alimentación') {
            $tabla = 'geocomidas';
        }
        $id = $punto->consultaId($tabla);
        $alojamiento = new Alojamientos();
        $alojamiento->setDpa_prov('04');
```

```
        $idCanton=$punto->consultaCantonId($canton);
        $tipoP=$punto->consultaIdTipo($tipo,$tabla);
        $alojamiento->setDpa_canton($idCanton);
        $alojamiento->setDpa_parr('000000');
        $alojamiento->setId($id);
        $alojamiento->setTipo($tipoP);
        $alojamiento->setFol($nroFol);
        $alojamiento->setEstablecimiento($establecimiento);
        $alojamiento->setPropietario($propietario);
        $alojamiento->setDireccion($direccion);
        $alojamiento->setCat_alojamiento($categoria);
        $alojamiento->setNro_hab($habitaciones);
        $alojamiento->setNro_plaza($plaza);
        $alojamiento->setPrecio_ocup_simple($precio);
        $alojamiento->setNro_mesas_com($mesas);
        $alojamiento->setNro_plazas_com($plazasComedor);
        $alojamiento->setNro_per_ocu($personal);
        $alojamiento->setTelefono($telefono);
        $alojamiento->setLatitud($lat);
        $alojamiento->setLongitud($lon);
        $alojamiento->setUrl($enlace);
        $mensaje=$punto->cargaPunto($alojamiento, $tabla);
        //$mensaje= $alojamiento->getDireccion();
        return $mensaje;

    }
}
$server = new soap_server;
$server->soap_defencoding = 'UTF-8';
```

```
$server->decode_utf8 = false;
$server->encode_utf8 = true;
$server->configureWSDL("Turismo", "http://turismoserver/mapas/util/ws");
$server->wsdl->schemaTargetNamespace = 'http://turismoserver/mapas/util/ws/xsd/';
$server->register("Servicios.crearTurismo", array("canton" => "xsd:string",
  "tipo" => "xsd:string",
  "tabla" => "xsd:string",
  "nombre" => "xsd:string",
  "direccion" => "xsd:string",
  "latitud" => "xsd:string",
  "longitud" => "xsd:string",
  "url" => "xsd:string"),
   array("return" => "xsd:string"), "http://turismoserver/mapas/util/ws",
"http://turismoserver/mapas/util/ws#crearTurismo", "rpc", "encoded", "Crear punto");
$server->register("Servicios.crearAlojamiento", array("tabla" => "xsd:string",
  "nroFol" => "xsd:string",
  "establecimiento" => "xsd:string",
  "propietario" => "xsd:string",
  "direccion" => "xsd:string",
  "habitaciones" => "xsd:string",
  "plaza" => "xsd:string",
  "precio" => "xsd:string",
  "mesas" => "xsd:string",
  "plazaComedor" => "xsd:string",
  "personal" => "xsd:string",
  "telefono" => "xsd:string",
  "enlace" => "xsd:string",
  "canton" => "xsd:string",
  "tipo" => "xsd:string",
```

```
  "categoria" => "xsd:string",
  "latitud" => "xsd:string",
  "longitud" => "xsd:string"),
   array("return" => "xsd:string"), "http://turismoserver/mapas/util/ws",
"http://turismoserver/mapas/util/ws#crearAlojamiento", "rpc", "encoded", "Crear punto
alojamiento");
$HTTP_RAW_POST_DATA = isset($HTTP_RAW_POST_DATA) ?
     $HTTP_RAW_POST_DATA : '';
$server->service($HTTP_RAW_POST_DATA);
```

ANEXO 5. GENERANDO LOS SERVICIOS WEB EN EL SERVIDOR II

- Agregar al archivo de gradle la dependencia que se requiere.

```
compile files('libs/ksoap2-android-assembly-2.4-jar-with-dependencies.jar')
```

- Instanciar las variables requeridas por el servicios web para su operación en la actividad requerida en este caso el de **TitleACtivity.java.**

```
private final String NAMESPACE = "http://"+servidor+"/webservice/";

private final String URL = "http://"+servidor+"/webservice/server.php";

//soap action de ws login

private final String SOAP_ACTION = "http://"+servidor+"/webservice#login";

//soap action de ws registrar

private final String SOAP_ACTION2 = "http://"+servidor+"/webservice#register";

private final String METHOD_NAME = "app.login";

private final String METHOD_NAME2 = "app.register";
```

- Para el caso mostrado se va a lanzar en el listener en el clic del botón registrar del método **onCreate()** con el siguiente contenido una tarea asíncrona que permita que la aplicación siga trabajando hasta que se tenga la respuesta de la tarea.

```
accederButton.setOnClickListener(new View.OnClickListener() {

 @Override

 public void onClick(View v) {
```

```
    EditText usuarioET = (EditText) findViewById(R.id.usuario);
     EditText contraseniaET = (EditText) findViewById(R.id.contrasenia);
    usuario= usuarioET.getText().toString();
    contrasenia=md5(contraseniaET.getText().toString());
    AsyncCallWS task = new AsyncCallWS();
    task.execute();
 }
});
```

- Posteriormente se crea una inner class que asíncrona la cual se extiende de la clase **AsyncTask**

```
// acceso en segundo plano login
private class AsyncCallWS extends AsyncTask<String, Void, Void> {
 @Override
 protected Void doInBackground(String... params) {
    logear(usuario,contrasenia);
    return null;
 }
 @Override
 protected void onPostExecute(Void result) {
    try {
       Toast.makeText(getBaseContext(),mensaje, Toast.LENGTH_SHORT).show();
       if (mensaje.equals("OK")){
          //acceder a siguiente pagina y pasar usuario
```

```
            Intent intent = new Intent();
            intent.putExtra("usuario",usuario);
            intent.setClass(TitleActivity.context, MenuActivity.class);
            startActivity(intent);
        }
    }
    catch (Exception excepcion){
        Toast.makeText(getBaseContext(),"no se puede conectar al servidor",
Toast.LENGTH_LONG).show();
    }
 }
 @Override
 protected void onPreExecute() {
    Log.v(TAG, "onPreExecute");
    Toast.makeText(getBaseContext(),"Revisando base", Toast.LENGTH_LONG).show();
 }
 @Override
 protected void onProgressUpdate(Void... values) {
 }
}
```

La validación del usuario y contraseña en este caso se realizan en una función que se encuentra en la clase principal **TitleActivity.java** con el nombre de **logear(),** que se muestra a continuación.

```
//funcion de validacion
public void logear(String usuario, String contrasenia) {
 //Create request
 SoapObject request = new SoapObject(NAMESPACE, METHOD_NAME);
 //Property which holds input parameters
```

```
PropertyInfo usuarioPI = new PropertyInfo();
PropertyInfo contraseniaPI = new PropertyInfo();
//Set Name
usuarioPI.setName("username");
usuarioPI.setValue(usuario);
usuarioPI.setType(String.class);
contraseniaPI.setName("password");
contraseniaPI.setValue(contrasenia);
contraseniaPI.setType(String.class);
//Add the property to request object
request.addProperty(usuarioPI);
request.addProperty(contraseniaPI);
 //Create envelope
SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(
      SoapEnvelope.VER11);
envelope.dotNet = true;
 //Set output SOAP object
envelope.setOutputSoapObject(request);
//Create HTTP call object
HttpTransportSE androidHttpTransport = new HttpTransportSE(URL);
try {
   //Invole web service
   androidHttpTransport.call(SOAP_ACTION, envelope);
   //Get the response
   Object response = (Object) envelope.getResponse();
   //Assign it to fahren static variable
   mensaje = response.toString();
} catch (Exception e) {
   e.printStackTrace();
```

```
 }
}
```

Dentro de la función se formatean las entradas requeridas por el servicios web, orientando a los requerimientos de SOAP, generando un objeto SOAP al cual se le agregan las propiedades requeridas por el servicio web.

- Las respuesta de este proceso define el acceso a la aplicación, en el caso de ser correcto o manda un mensaje de tipo **Toast** en el caso de ser incorrecto, dentro de la clase asíncrona en el método **onPostExecute()** como se muestra a continuación.

```
@Override
protected void onPostExecute(Void result) {
 try {
    Toast.makeText(getBaseContext(),mensaje, Toast.LENGTH_SHORT).show();
    if (mensaje.equals("OK")){
       //acceder a siguiente pagina y pasar usuario
       Intent intent = new Intent();
       intent.putExtra("usuario",usuario);
       intent.setClass(TitleActivity.context, MenuActivity.class);
       startActivity(intent);
    }
 }
```

```
 catch (Exception excepcion){

    Toast.makeText(getBaseContext(),"no se puede conectar al servidor",
Toast.LENGTH_LONG).show();

  }

}
```

ANEXO 6. MANUAL DE USUARIO DE LA APLICACIÓN MÓVIL DE GEOREFEREMCIACIÓN

Para que la aplicación móvil funcione en un Smartphone y poder utilizarla para el levantamiento de información y georeferenciación se debe seguir los pasos que se detallan acontuación:

Paso1. Solicitar la autorización al encargado del manejo de la app en la Prefectura del Carchi para utilizar la app.

Paso 2. Instalar la app en el dispositivo móvil, recordando que su versión no debe ser inferior a 4.1.

Paso 3. Una vez instalada la aplicación se muestra una imagen de android en la pantalla del Smartphone, dar click para poder ingresar.

Paso 4. Registrar el usuario y el pasword

Paso 5. Dar click en loginque **s**erá la pantalla que da la bienvenida.

Paso 6. Una vez que se ha accedido a la aplicación se tiene acceso al menú de la misma que cuenta con tres opciones: Chat, Mapa, Form, donde se selecciona la opción reuqerida.

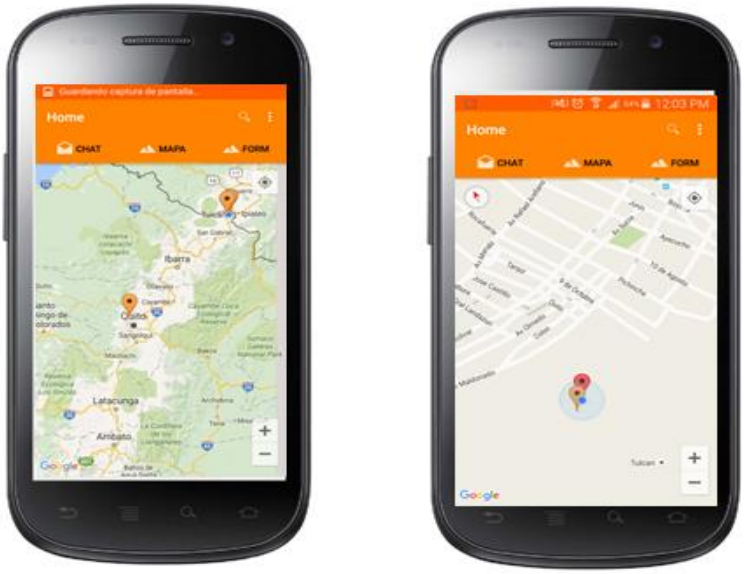
Paso 7. Si dá click en la opción chat debe ingresar nuevamente el usuario y la contraseña.

Paso 8. Seleccionar los usuarios con quien se va a interactuar

Paso 9. Una vez seleccionado el usuario se escribe el mensaje y se da click en enviar.

Paso 10. Si su desición es ya no enviar mas mensajes da click en salir.

Paso 11. Si selecciona la opción mapa, usted puede observar que se despliega un mapa en el cual se puede observar la ubicación de los usuarios de la aplicación asi como la ubicación del dispositivo que hace la petición, desplegando la información del usuario y la fecha de conexión.



Paso 12. Si selecciona la opción Form, aparece la ventana levantamiento de puntos, en la cual se selecciona la categoría de puntos.

Paso 13. Dar click en ver formulario, donde aparece un formulario de acurdo a la categoria de puntos selecionada en el paso anterior.

Paso 14. Seleccionar tipo y cantón, y se ingresa la información solicitada que va a ser levantada in situ.

Paso 15. Dar click en registrar donde se enviará la información al servidor para su procesamiento.